Linear Algebra and its Applications 574 (2019) 123–152

Contents lists available at ScienceDirect

Linear Algebra and its Applications

www.elsevier.com/locate/laa

Graphs with absorption: Numerical methods for the absorption inverse and the computation of centrality measures

Michele Benzi [a,*], Paraskevi Fika [b], Marilena Mitrouli [b]

[a] *Scuola Normale Superiore, Piazza dei Cavalieri 7, 56126 Pisa, Italy*
[b] *University of Athens, Department of Mathematics, Panepistimiopolis, 15784 Athens, Greece*

ARTICLE INFO

Article history:
Received 3 August 2018
Accepted 26 March 2019
Available online 3 April 2019
Submitted by R. Brualdi

MSC:
primary 15A09
secondary 65F20, 05C81, 94C15

Keywords:
Laplacian matrix
Group inverse
Absorption inverse
Centrality measure
Matrix factorizations
Krylov subspace methods
Preconditioning

ABSTRACT

The absorption inverse, studied in Jacobsen and Tien (2018) [13], is a generalized inverse specifically introduced for the analysis of graphs with absorption. In this paper we consider numerical methods for the efficient computation of the absorption inverse and related quantities. Both direct and iterative methods are developed. We also consider different centrality measures for graphs with absorption, as well as fast updating/downdating techniques. Numerical experiments show that computations on graphs with up to 36 million edges can be performed quickly on a standard laptop.

* Corresponding author.
E-mail addresses: michele.benzi@sns.it (M. Benzi), pfika@math.uoa.gr (P. Fika), mmitroul@math.uoa.gr (M. Mitrouli).

https://doi.org/10.1016/j.laa.2019.03.026
0024-3795/

1. Introduction

Let $G = (V, E, \mathbf{w})$ be a weighted graph, directed or undirected, and let $A \in \mathbb{R}^{n\times n}$ be the associated adjacency matrix whose non-zero entries a_{ij} correspond to the weight of the edge from node j to node i. Then, the graph Laplacian matrix $L \in \mathbb{R}^{n\times n}$ is given by $L = W - A$, where the matrix $W \in \mathbb{R}^{n\times n}$ is diagonal with each entry w_{ii} given by the sum of outgoing weights from the node i, i.e., $w_{ii} = \sum_{j=1}^{n} a_{ji}$ (weighted outdegree). It is well known that $\operatorname{rank}(L) \leq n-1$ and that equality holds if and only if the graph is strongly connected. If the graph is reducible, the adjacency matrix A (and hence L) can be permuted to block triangular form (block diagonal if the graph is undirected) with irreducible diagonal blocks, one for each strongly connected component of the graph; the considerations and algorithms in this paper can then be applied to each diagonal block separately. Therefore, unless otherwise specified, we will always assume that G is strongly connected.

We caution the reader that the convention "$a_{ij} \neq 0$ if there is an edge from node j to node i" differs from that adopted in most papers and books on graph theory. Here we adopt this convention in order to be consistent with the notation used by the authors of [13], on whose work this paper builds. The convention implies that for any graph, L has zero column sums.

Following [13], we consider *graphs with absorption*, i.e., we suppose that each node of the graph represents a transient state in a Markov process, and that each transient state comes with a transition rate $d_i > 0$ to an absorbing state (labeled $n+1$). Such graphs arise naturally in many applications, for example in the modeling of disease spreading in community networks [22]. Let $\mathbf{d} = [d_1, \ldots, d_n]^T$ be the vector of absorption rates. The resulting graph with absorption is denoted by $(G, \mathbf{d})$. Let now L be the Laplacian matrix associated with G. The authors of [13] introduced a generalized inverse of L that captures much valuable information about transient random walks on the graph by combining the known node absorption rates d_i with the structural properties of the underlying graph G encoded in the Laplacian matrix L. The definition and basic properties of the absorption inverse are given in Section 2. Among other things, the absorption inverse can be used to define a notion of distance, as well as a centrality measure for ranking the nodes in an absorption graph. Furthermore, the absorption inverse can be used for graph partitioning purposes. The authors of [13] left open the issue of devising efficient computational approaches for computing the absorption inverse or quantities associated with it, such as the centrality vector. In this paper we address the efficient computation of these quantities, as well as other computational aspects, including the problem of how to perform cheap updates of the quantities of interest when the underlying absorption graph undergoes a modification (such as the addition/deletion of an edge). In addition, we propose alternative centrality measures based on the absorption inverse, which give more weight to the graph topology (compared to the measure proposed in [13]) while still taking into account the absorption rates associated with the graph nodes.

The remainder of the paper is organized as follows. In Section 2 we provide some basic definitions and facts about the absorption inverse. Section 3 is devoted to a discussion of centrality measures based on the absorption inverse. The core of the paper, Section 4, is devoted to a description of direct and iterative methods for the accurate and efficient computation of the absorption inverse and related quantities. The updating/downdating problem and techniques for estimating individual entries of the absorption inverse are studied in Sections 5 and 6, respectively. Numerical experiments illustrating the performance of the various algorithms on both undirected and directed graphs with absorption are presented in Section 7. Finally, Section 8 contains some conclusive remarks.

Throughout the paper, the superscript T denotes the transpose and $\mathbf{e}_i$ stands for the ith column of the identity matrix of dimension n, denoted I_n. We denote by $\mathbf{1}$ and $\mathbf{0}$ the column vectors of length n of all ones and of all zeros, respectively. The $m \times n$ matrix with all zeros is denoted by $\mathbf{0}_{m,n}$.

2. Preliminaries and background

Recall that a matrix X^- is said to be a $\{1,2\}$-inverse of a matrix X if it satisfies the first two of the Penrose conditions, namely

$$XX^-X = X, \qquad X^-XX^- = X^-, \tag{1}$$

see [6]. We also recall that a matrix that satisfies the first (but not necessarily the second) condition in (1) is called a $\{1\}$-inverse of the matrix X.

The *absorption inverse of L with respect to* $\mathbf{d}$, denoted as L^d, is a $\{1,2\}$-inverse X of L which satisfies the following conditions:

$$\begin{aligned} XL\mathbf{y} &= \mathbf{y}, \quad \text{for } \mathbf{y} \in N_{1,0}, \\ X\mathbf{y} &= \mathbf{0}, \quad \text{for } \mathbf{y} \in R_{1,0}, \end{aligned}$$

where

$$\begin{aligned} N_{1,0} &= \{\mathbf{x} \in \mathbb{R}^n : D\mathbf{x} \in \mathrm{Range}(L)\}, \\ R_{1,0} &= \{D\mathbf{x} : \mathbf{x} \in \mathrm{Ker}(L)\}, \end{aligned}$$

and D is the diagonal matrix whose entries are the absorption rates $d_1, d_2, \ldots, d_n$. We write $X = L^d$. Various representations and properties of L^d can be found in [13]. For later use, we list here three basic results from [13].

Proposition 1. *[13, Theorem 2] Let $(G, \mathbf{d})$ be a strongly connected graph with positive absorption vector $\mathbf{d}$. Then L^d exists and is unique.*

Proposition 2. *[13, Theorem 3] Under the same assumptions of Proposition 1, the following identity holds:*

$$L^d = (I_n - VD)Y(I_n - DV), \tag{2}$$

where Y is any $\{1\}$-inverse of the Laplacian matrix L, $V = \mathbf{v}\mathbf{1}^T/\tilde{d}$, $\mathbf{v}$ is a positive vector in $\mathrm{Ker}(L)$ *with* $\sum_{i=1}^n v_i = 1$, *and* $\tilde{d} = \mathbf{d}^T\mathbf{v}$.

We emphasize that L^d, being unique, does not depend on the choice of the $\{1\}$-inverse Y (there are infinitely many such generalized inverses of L). We also note that the Perron–Frobenius theorem guarantees the existence of a unique positive vector $\mathbf{v} \in \mathrm{Ker}(L)$ with $\sum_{i=1}^n v_i = 1$.

Proposition 3. *[13, Proposition 2] Under the same assumptions of Proposition 1, the following identity holds:*

$$L^d = D^{-1}(LD^{-1})^{\#}, \tag{3}$$

where $(LD^{-1})^{\#}$ is the group inverse of LD^{-1}.

We recall that the group generalized inverse of a square matrix A of index one is the unique matrix $A^{\#}$ such that $AA^{\#}A = A$, $A^{\#}AA^{\#} = A^{\#}$ and $AA^{\#} = A^{\#}A$. See [6] for further details.

Below we prove two simple additional facts about the absorption inverse.

Remark 1. It is a simple consequence of the definition that the absorption inverse L^d is unaffected under scaling of the absorption rates. This implies that in a graph with absorption, a node is characterized by its absorption rate relative to that of the others; i.e., only ratios d_i/d_j (with $j \neq i$) are meaningful, not the actual values d_i and d_j. Therefore, a graph of type $(G, \mathbf{1})$ is equivalent to a "standard" graph, in the sense that only the connectivity properties of the nodes matter, and not their relation to the absorbing state of the underlying Markov process. In the following, we will refer to this case as the "equal absorption" case. Note that for such a graph we have $L^d = L^{\#}$ by Proposition 3. Thus, as observed in [13], the absorption inverse can be regarded as a generalization of the group inverse to the case of graphs with absorption.

Proposition 4. *The matrices LL^d and L^dL are similar under the diagonal transformation D.*

Proof. We give two proofs of this fact.

(1) From Theorem 1 and Lemma 1 in [13] we have that $L^dL = I_n - VD$ and $LL^d = I_n - DV$, hence $LL^d = DL^dLD^{-1}$ since $LL^dD = (I_n - DV)D = D - DVD = D(I_n - VD) = DL^dL$.
(2) By Proposition 3 we have that $L^d = D^{-1}(LD^{-1})^{\#}$. Then using the third property of the group inverse and the equality $(LD^{-1})^{\#} = DL^d$, we have that the rela-

tion $(LD^{-1})^{\#}(LD^{-1}) = (LD^{-1})(LD^{-1})^{\#}$ implies $DL^d(LD^{-1}) = (LD^{-1})DL^d$, i.e., $DL^dLD^{-1} = LL^d$. □

In the next section we discuss the use of the absorption inverse to define node centrality measures for graphs with absorption.

3. Graph centrality measures

One of the fundamental problems in the study of real-world networks is the identification of the most important nodes in the network [7,19]. Since the notion of importance is clearly context-dependent, it is not surprising that many different notions of centrality have been proposed in the literature. A centrality measure specifically designed for graphs with absorption has been proposed in [13]. We recall that a (possibly weighted) directed graph is said to be *balanced* if for every node i in the graph, the indegree equals the outdegree: $\sum_{j=1}^{n} a_{ij} = \sum_{j=1}^{n} a_{ji}$. Note that in this case $\mathbf{1}^T L = \mathbf{0}_{1,n}$ and $L\mathbf{1} = \mathbf{0}_{n,1}$. For a balanced, strongly connected graph with absorption the authors of [13] propose to rank the nodes using the entries of the vector $L^d\mathbf{1}$; that is, the centrality score of the ith node is given by the ith row sum of the absorption inverse L^d. The reason for this specific notion of centrality is grounded in the probabilistic interpretation of the entries of L^d and is related to the behavior of the absorbing random walk associated to $(G, \mathbf{d})$. Roughly speaking, a node which is "near" the absorbing state of the random walk has low centrality, since the system being considered is likely to spend little time in such a state. Conversely, a high centrality node will be "far" from the absorbing state and therefore the system is likely to spend a relatively large fraction of time in the corresponding state. We refer to [13] for a more precise discussion.

As noted by the authors of [13], the entries of $L^d\mathbf{1}$ can be negative, which is contrast with virtually all other known centrality measures. This, however, is only a minor drawback: in fact, the resulting ranking of the nodes is still meaningful, with the node with the lowest (i.e., leftmost) centrality score being ranked at the bottom, the second lowest just above it, and so on. The authors of [13] argue that the centrality measure $L^d\mathbf{1}$ takes into account both the graph structure and the absorption rates. However, when the absorption rates are all equal it holds that $L^d\mathbf{1} = \mathbf{0}$. To see this, note that for $D = I_n$ we have $L^d = (I_n - V)Y(I_n - V)$ and for a balanced graph

$$(I_n - V)\mathbf{1} = \mathbf{1} - V\mathbf{1} = \mathbf{1} - \frac{\mathbf{v}\mathbf{1}^T}{\mathbf{1}^T\mathbf{v}}\mathbf{1} = \mathbf{1} - \mathbf{1} = \mathbf{0},$$

since $\mathbf{v} = \frac{1}{n}\mathbf{1}$. Therefore, when the absorption rates are all equal, this measure is unable to distinguish between the nodes in the graph. For example (as noted by the authors), all nodes in any star graph are given the same centrality score. Hence, in the special case when all the absorption rates become equal, the centrality measure based on $L^d\mathbf{1}$ does not reduce to any known centrality measure for "standard" graphs, since all such

measures would rank the hub of the star graph as the most central node, with all the other nodes being tied for a (distant) second place. By continuity, if the absorption rates are all different but close to one another, the centrality scores will also be close and the centrality measure will have difficulties differentiating between nodes, regardless of the graph topology. This suggests that the proposed centrality measure gives relatively little weight to the graph topology, with the absorption rates being far more important for the ranking of the nodes.

In [13], the authors left open the possibility of defining alternative centrality measures for graphs with absorption. In this section we introduce some modifications to the measure proposed in [13] that aim at giving more weight to the topology of the graph, so that in the limit of equal absorption rates the centrality measure is still able to discriminate between nodes for most types of graph. We also consider centrality measures that are able to take into account the dual role each node plays in a directed graph, namely, that of a "broadcaster" (or "hub") and that of "receiver" (or "authority"). While in a balanced graph this distinction is less crucial (and disappears in the special case of undirected graphs), here we would like to drop the assumption of balancedness and therefore it becomes important to be able to account for both roles.

As a motivating example, consider a directed graph representing a communication or information network. Suppose that in this graph there is an absorbing node, labeled $n+1$. This is also called a "dangling" node or "sink": information can reach this node, but the node cannot communicate it to any other node. Nodes 1 through n are assigned prescribed absorption rates. It is clear that in such a graph, topology plays an important role, and any centrality measure must properly account for the role each node plays in the robustness and efficiency of information flow on the network. This requires centrality indices that strike a balance between the distance from the absorbing state and the structural properties of the graph G.

Our first proposal is to replace $L^d\mathbf{1}$ with the vector $L^dW\mathbf{1} = L^d\mathbf{w}$, where $\mathbf{w} = \mathrm{diag}(W)$. This variant is similar to other centrality measures, such as the use of a "personalization vector" in Google's PageRank algorithm [16] or the well-known Katz index [15], which can be written (for a weighted undirected graph) in the form $(I_n - \alpha A)^{-1}A\mathbf{1} = (I - \alpha A)^{-1}\mathbf{w}$, with $0 < \alpha < 1/\|A\|_2$.

As we shall see, this centrality measure is able to discriminate between nodes except for very special situations such as when the graph is balanced and the diagonal matrix W is proportional to the absorption matrix D, i.e., $W = \mu D$ for some $\mu > 0$. In this case we have

$$L^dW\mathbf{1} = \mu L^dD\mathbf{1} = \mathbf{0}. \tag{4}$$

Equation (4) follows from

$$L^dD\mathbf{1} = (I_n - VD)Y(I_n - DV)D\mathbf{1}$$

and the fact that

$$(I_n - DV)D\mathbf{1} = (D - DVD)\mathbf{1} = D\mathbf{1} - D\mathbf{1} = \mathbf{0}.$$

Indeed, we have

$$VD\mathbf{1} = \frac{\mathbf{v}\mathbf{1}^T\mathbf{d}}{\mathbf{d}^T\mathbf{v}} = \frac{(1/n)\mathbf{1}^T\mathbf{d}\mathbf{1}}{(1/n)\mathbf{d}^T\mathbf{1}} = \mathbf{1},$$

due to the already observed fact that $v = (1/n)\mathbf{1}$ for a balanced graph.

The case of $W = \mu D$ is highly unlikely to occur in practice. For instance, it can happen in the following cases:

1. All absorptions are equal and the weight matrix W is proportional to the identity, for example in a cycle graph C_n;
2. In a star graph S_{n-1}, when the absorption rates are all equal except from the central node that has absorption rate equal to $n-1$ times the others' rate.

Obviously, the measure $L^dW\mathbf{1}$ gives the same node ranking as $L^d\mathbf{1}$ when the weighted outdegrees of the nodes are all equal, i.e., when $W = \mu I_n$, such as in the case of the (unweighted) cycle graph C_n.

An alternative centrality measure can be obtained from the diagonal entries of the matrix L^dW. As argued in [13, page 137], the entry L^d_{ij} of L^d gives an indication of the time spent in vertex i in a transient random walk starting at vertex j.[1] Since a transient random walk that starts at an important (highly central) node is likely to return often to that node, it is reasonable to take L^d_{ii} as a measure of the centrality of node i in $(G, \mathbf{d})$. We point out here the resemblance of this measure with *subgraph centrality* [8], which consists of taking the diagonal entries of some function of the adjacency matrix, such as the matrix exponential or the resolvent; this corresponds to taking weighted sums of the number of walks on the graphs that visit node i, with longer walks being given a smaller weight.

An advantage of this index is that for a strongly connected graph, it is guaranteed to be positive (see [13, Corollary 4]). However, it suffers from the same potential drawback of the row sums of L^d, namely, it tends to discount the role of node connectivity within the underlying weighted graph. Similar to our previous variant, we propose instead to take the diagonal entries of L^dW as the centralities of the corresponding nodes. This simply amounts to multiplying the diagonal entries of L^d by the weighted outdegrees of the corresponding nodes. Clearly, the resulting centrality measure is strictly positive. Also, it is able to discriminate between nodes in certain situations where $L^dW\mathbf{1}$ cannot.

[1] Strictly speaking, this interpretation is valid provided that absorption is slow relative to transitions between vertices, i.e., $\|D\| \ll \|L\|$. However, Remark 1 implies that we can always rescale the entries of D without affecting the entries of L^d, therefore we can always assume this assumption to be satisfied.

Table 1
Centrality scores of the star graph with absorption rates $\mathbf{d} = [1, 2, 0.1, \ldots, 0.1]^T$ (Case 1), $\mathbf{d} = [0.2, 0.1, \ldots, 0.1, 0.2]^T$ (Case 2) and $\mathbf{d} = \mathbf{1}$ (Case 3).

Node	Case 1			Case 2		Case 3
	$L^dW\mathbf{1}$	$L^d\mathbf{1}$	$\mathrm{diag}(L^dW)$	$L^dW\mathbf{1}$	$L^d\mathbf{1}$	$L^dW\mathbf{1}$
1	1.54e0	9.09e-1	8.40e-1	−1.11e0	−5.56e-1	−1.02e-1
2	−1.89e0	−1.09e0	2.69e-1	2.22e-1	2.22e-1	−1.02e-1
3	4.62e0	2.71e0	1.35e0	2.22e-1	2.22e-1	−1.02e-1
4	4.62e0	2.71e0	1.35e0	2.22e-1	2.22e-1	−1.02e-1
5	4.62e0	2.71e0	1.35e0	2.22e-1	2.22e-1	−1.02e-1
6	4.62e0	2.71e0	1.35e0	2.22e-1	2.22e-1	−1.02e-1
7	3.97e0	1.91e0	2.47e0	5.56e-1	0	6.12e-1

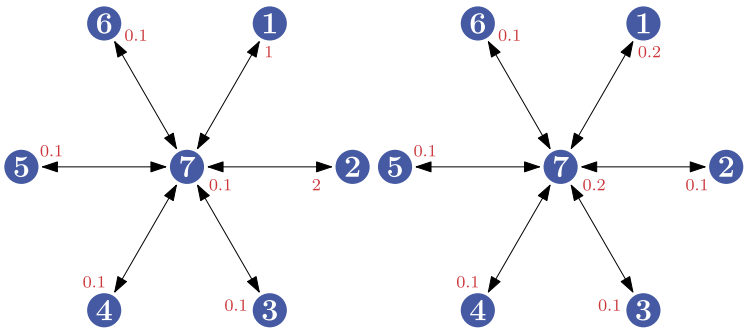

Fig. 1. The star graph S_6 with absorptions rates $\mathbf{d} = [1, 2, 0.1, \ldots, 0.1]^T$ (left) and $\mathbf{d} = [0.2, 0.1, \ldots, 0.1, 0.2]^T$ (right).

A disadvantage of this measure is that for very large graphs it is generally much more expensive to compute than that based on $L^dW\mathbf{1}$.

In the case of unbalanced graphs, we consider as centrality measure the quantity $L^dW_s\mathbf{1}$, where $W_s = W_o + W_i$, W_o is the diagonal matrix with the weighted outdegrees (its ith diagonal entry being given by $\sum_{j=1}^n a_{ji}$), and W_i is the diagonal matrix with the weighted indegrees (its ith diagonal entry being equal to $\sum_{j=1}^n a_{ij}$). Note that this yields an overall centrality ranking of the nodes, but it does not differentiate between the two roles (broadcaster and receiver) a node plays in a directed graph. Therefore we also consider the measures $L^dW_o\mathbf{1}$ and $L^dW_i\mathbf{1}$ for ranking hubs and authorities, respectively. Along the same lines as in the balanced case, if the matrix D is proportional to the identity and the same holds for any of the matrices W_s, W_o and W_i, then the corresponding centrality measures are uninformative.

Example 1. Let us first consider the (unweighted, undirected) star graph S_6, with six peripheral nodes and the 7th node as the center node. This graph was also used in the numerical examples in [13]. In Table 1 we report the centrality scores of each node using the $L^dW\mathbf{1}$ and $L^d\mathbf{1}$ centrality measures, for two different sets of absorption rates (see Fig. 1). In the last column of the table, the measure $L^dW\mathbf{1}$ is tested for the case of all equal absorptions ($D = I_n$). We recall that in this case $L^d\mathbf{1} = \mathbf{0}$, therefore the measure proposed in [13] is unable to differentiate the center node from all the other nodes (thus

we do not include it in the tables). While this is justifiable in terms of the fact that each node in S_6 represents a transient state with equal absorption rate to the absorbing state (in the underlying absorbing random walk), it may be desirable to also take into account the obviously prominent role the center node plays in the topology of S_6. Put differently, the measure $L^dW\mathbf{1}$ can be regarded as using a somewhat different metric (as compared to $L^d\mathbf{1}$) when measuring node distances from the absorbing state.

From Table 1 we can see that in the first case, where the center node has the same absorption rate ($= 0.1$) as the peripheral nodes $3, 4, 5, 6$ while nodes 1 and 2 have absorption rates 1 and 2, respectively, the $L^dW\mathbf{1}$ measure ranks the center node as the second most important, behind the nodes $3, 4, 5, 6$ (all given equal rank); the nodes 1 and 2 are ranked as the least important ones. Likewise, the $L^d\mathbf{1}$ measure ranks the nodes $3, 4, 5, 6$ as the most important, followed by the center node; again, nodes 1 and 2 are ranked as the least important nodes. In contrast, using the diagonal entries of L^dW results in the center node being ranked as the most important one, followed by nodes $3, 4, 5, 6$ (all given equal rank), with nodes 1 and 2 again being ranked as the least important. It is also worth noting that for star graphs with more than 30 nodes, the center node becomes the most central node also according to $L^dW\mathbf{1}$, when the absorption rates are again $\mathbf{d} = [1, 2, 0.1, \ldots, 0.1]^T$. This shows that incorporating the weighted degree matrix W in the centrality measure results in the connectivity of the graph playing a larger role than with $L^d\mathbf{1}$.

In the second case, where the center node has the same absorption rate as node 1 (equal to 0.2) and all the other peripheral nodes have absorption rates equal to 0.1, the measure $L^dW\mathbf{1}$ ranks the node 7 as the most important, whereas the measure $L^d\mathbf{1}$ ranks the peripheral nodes $2 - 6$ as the most important (all with the same score), followed by the center node and finally by node 1. The same ranking is also obtained using the diagonal entries of L^dW (not shown); however, as in case 1, when the number of nodes $n + 1$ is large enough, the center node in S_n becomes the most important.

We observe that when using $L^dW\mathbf{1}$ to rank nodes in a n-node star graph in which the absorption rates are equal for all nodes except for the center node, this remains the most important one as long as its absorption rate is less than $n - 1$ times that of the others, whereas it becomes the least important one when its absorption rate is greater than $n-1$ times the others' rate. As already mentioned, the measure is unable to discriminate among nodes when the absorption rate of the center node is exactly $n - 1$ times that of the other nodes.

Finally, in the last column of Table 1 we see that the centrality scores obtained using $L^dW\mathbf{1}$ are the expected ones in case of all equal absorptions.

Example 2. Let us consider a path graph and a cycle graph with 8 nodes, again with all weights equal to 1 (see Fig. 2). In Table 2 we present the centrality scores of their nodes using the $L^dW\mathbf{1}$ and $L^d\mathbf{1}$ centrality measures. In the cycle graph, for the absorption rates $\mathbf{d} = \mathbf{1}$, it holds $W = 2D$ and thus $L^dW\mathbf{1} = \mathbf{0}$. Therefore, the measure $\mathrm{diag}(L^dW)$ is used instead.

Table 2
Centrality scores of the path graph and the cycle graph with absorption rates $\mathbf{d} = \mathbf{1}$ (Case 1) and $\mathbf{d} = [1, 1, 2, 1, \ldots, 1]^T$ (Case 2).

Node	Path			Cycle		
	Case 1	Case 2		Case 1	Case 2	
	$L^dW\mathbf{1}$	$L^dW\mathbf{1}$	$L^d\mathbf{1}$	diag(L^dW)	$L^dW\mathbf{1}$	$L^d\mathbf{1}$
1	−8.75e-1	−1.63e0	−4.07e-1	1.31e0	2.96e-1	1.48e-1
2	−1.25e-1	−1.07e0	−5.19e-1	1.31e0	−2.59e-1	−1.30e-1
3	3.75e-1	−9.63e-1	−7.41e-1	1.31e0	−1.04e0	−5.19e-1
4	6.25e-1	2.59e-1	−1.85e-1	1.31e0	−2.59e-1	−1.30e-1
5	6.25e-1	1.04e0	2.59e-1	1.31e0	2.96e-1	1.48e-1
6	3.75e-1	1.37e0	5.93e-1	1.31e0	6.30e-1	3.15e-1
7	−1.25e-1	1.26e0	8.15e-1	1.31e0	7.41e-1	3.70e-1
8	−8.75e-1	7.04e-1	9.26e-1	1.31e0	6.30e-1	3.15e-1

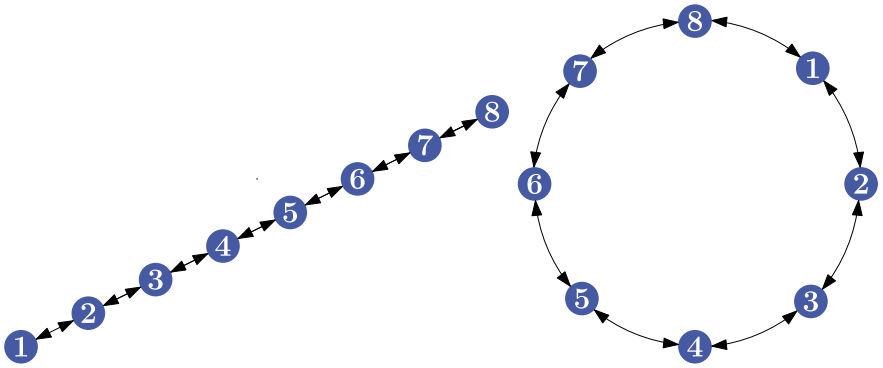

Fig. 2. The 8-node path graph (left) and the 8-node cycle graph (right).

In Table 2 we notice that in the case of all equal absorptions, the measure $L^dW\mathbf{1}$ ranks the nodes 4 and 5 as the most important in the path graph. In the case of the cycle graph, the measure diag(L^dW) gives the same centrality scores for all nodes. This is as expected. In case that all nodes have the same absorption rate equal to 1, except for node 3 that has absorption rate 2, we observe the following. In case of the path graph, the $L^dW\mathbf{1}$ measure ranks the node 6 as the most important, whereas the $L^d\mathbf{1}$ measure ranks the node 8 as the most important. This happens because this measure takes into account the distance from the highly absorbing node. Changing the absorption rate of node 3 to any value > 1 (even for 1.000001 or smaller) the measure $L^d\mathbf{1}$ still gives the node 8 as the most important, whereas, the measure $L^dW\mathbf{1}$ is being "adjusted". For instance, testing the measure $L^dW\mathbf{1}$ by setting $\mathbf{d} = [1, 1, 3, 1, \ldots, 1]^T$, the node 7 becomes the most important and for $d_3 > 7$ the node 8 becomes the most important. Actually, for any n-nodes path graph, when the absorption rates are all equal except for one node, say node i, one of the two peripheral nodes becomes the most "significant" when the absorption rate of node i is greater than $n-1$ times the others' rate. In case of $d_i = n-1$, then the nodes 1 and 2 have the same centrality scores, as well as nodes n and $n-1$. In case of the cycle graph, the two measures have the same behavior, ranking node 7 as

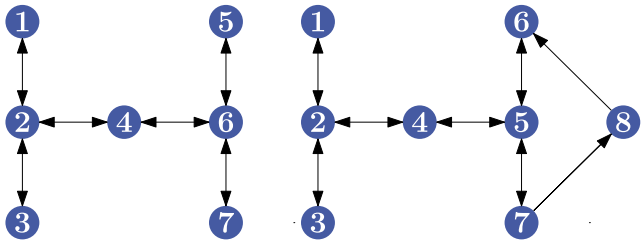

Fig. 3. A 7-node balanced graph (left) and a 8-node unbalanced graph (right).

Table 3
Centrality scores of the graph of Fig. 3 (left) with absorption rates $\mathbf{d} = \mathbf{1}$ (Case 1), $d_i = 1$, $i = 1, \ldots, 8$, $i \neq 4$ and $d_4 = 1.1$ (Case 2), $d_4 = 2$ (Case 3) and $d_4 = 3$ (Case 4).

Node	Case 1	Case 2		Case 3		Case 4	
	$L^dW\mathbf{1}$	$L^dW\mathbf{1}$	$L^d\mathbf{1}$	$L^dW\mathbf{1}$	$L^d\mathbf{1}$	$L^dW\mathbf{1}$	$L^d\mathbf{1}$
1	−3.27e-1	−3.12e-1	1.27e-2	−1.25e-1	1.56e-1	1.48e-1	3.46e-1
2	3.88e-1	3.78e-1	−1.39e-3	3.75e-1	3.12e-2	4.81e-1	1.23e-1
3	−3.27e-1	−3.12e-1	1.27e-2	−1.25e-1	1.56e-1	1.48e-1	3.46e-1
4	5.31e-1	4.48e-1	−4.36e-2	−1.25e-1	−3.44e-1	−5.19e-1	−5.43e-1
5	−3.27e-1	−3.12e-1	1.27e-2	−1.25e-1	1.56e-1	1.48e-1	3.46e-1
6	3.88e-1	3.78e-1	−1.39e-3	3.75e-1	3.12e-2	4.81e-1	1.23e-1
7	−3.27e-1	−3.12e-1	1.27e-2	−1.25e-1	1.56e-1	1.48e-1	3.46e-1

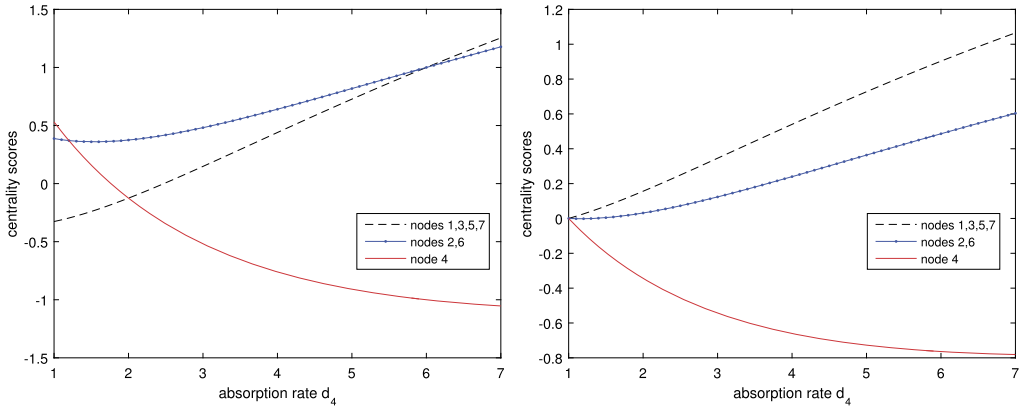

Fig. 4. The centrality scores $L^dW\mathbf{1}$ (left) and $L^d\mathbf{1}$ (right) as the absorption rate of node 4 increases.

the most important, since the two measures give proportional centrality scores, because $W = 2I_n$.

Example 3. Let us consider the (unweighted) graph depicted in Fig. 3 (left). In Table 3 we present the centrality scores of its nodes using the $L^dW\mathbf{1}$ and $L^d\mathbf{1}$ centrality measures. In Case 1 we consider absorption rates $\mathbf{d} = \mathbf{1}$, whereas in Cases 2, 3 and 4 we consider $d_i = 1$, $i = 1, \ldots, 7$, $i \neq 4$, and $d_4 > 1$.

The ranking of the nodes that the centrality scores of Table 3 yield, is also depicted in Fig. 4. We notice that in the case of all equal absorptions, the measure $L^dW\mathbf{1}$ ranks

Table 4
Centrality scores of the graph of Fig. 3 (right) with absorption rates $\mathbf{d} = \mathbf{1}$ (Case 1).

Node	Case 1			
	$L^d W_s \mathbf{1}$	$L^d W_o \mathbf{1}$	$L^d W_i \mathbf{1}$	$L^d \mathbf{1}$
1	−2.34e0	−1.20e0	−1.14e0	−4.31e-1
2	−6.09e-1	−3.38e-1	−2.71e-1	−3.64e-1
3	−2.34e0	−1.20e0	−1.14e0	−4.31e-1
4	5.91e-1	2.62e-1	3.29e-1	−1.64e-1
5	1.52e0	7.29e-1	7.96e-1	1.02e-1
6	3.87e-1	−1.07e-1	4.93e-1	2.53e-1
7	1.33e0	8.98e-1	4.31e-1	2.84e-1
8	1.46e0	9.64e-1	4.98e-1	7.51e-1

Table 5
Centrality scores of the graph of Fig. 3 (right) with absorption rates $d_i = 1$, $i = 1, \ldots, 8,\ i \neq 4$ and $d_4 = 1.5$ (Case 2), $d_4 = 2$ (Case 3).

Node	Case 2				Case 3			
	$L^d W_s \mathbf{1}$	$L^d W_o \mathbf{1}$	$L^d W_i \mathbf{1}$	$L^d \mathbf{1}$	$L^d W_s \mathbf{1}$	$L^d W_o \mathbf{1}$	$L^d W_i \mathbf{1}$	$L^d \mathbf{1}$
1	−2.22e0	−1.14e0	−1.08e0	−3.75e-1	−2.02e0	−1.04e0	−9.83e-1	−3.01e-1
2	−7.19e-1	−3.91e-1	−3.28e-1	−3.75e-1	−7.30e-1	−3.94e-1	−3.36e-1	−3.60e-1
3	−2.22e0	−1.14e0	−1.08e0	−3.75e-1	−2.02e0	−1.04e0	−9.83e-1	−3.01e-1
4	−2.19e-1	−1.41e-1	−7.81e-2	−3.75e-1	−8.48e-1	−4.53e-1	−3.94e-1	−5.36e-1
5	1.53e0	7.34e-1	7.97e-1	1.25e-1	1.62e0	7.82e-1	8.41e-1	1.70e-1
6	9.22e-1	1.64e-1	7.58e-1	4.38e-1	1.52e0	4.67e-1	1.06e0	6.37e-1
7	1.39e0	9.30e-1	4.61e-1	3.13e-1	1.49e0	9.79e-1	5.09e-1	3.49e-1
8	1.64e0	1.05e0	5.86e-1	8.13e-1	1.84e0	1.16e0	6.85e-1	8.79e-1

Table 6
Centrality scores of the graph of Fig. 3 (right) with absorption rates $d_i = 1$, $i = 1, \ldots, 8$, $i \neq 5$ and $d_5 = 1.5$ (Case 4), $d_5 = 2$ (Case 5).

Node	Case 4				Case 5			
	$L^d W_s \mathbf{1}$	$L^d W_o \mathbf{1}$	$L^d W_i \mathbf{1}$	$L^d \mathbf{1}$	$L^d W_s \mathbf{1}$	$L^d W_o \mathbf{1}$	$L^d W_i \mathbf{1}$	$L^d \mathbf{1}$
1	−1.45e0	−7.58e-1	−6.95e-1	−1.56e-1	−5.71e-1	−3.15e-1	−2.56e-1	1.14e-1
2	4.69e-2	−7.81e-3	5.47e-2	−1.56e-1	7.23e-1	3.32e-1	3.91e-1	5.54e-2
3	−1.45e0	−7.58e-1	−6.95e-1	−1.56e-1	−5.71e-1	−3.15e-1	−2.56e-1	1.14e-1
4	5.47e-1	2.42e-1	3.05e-1	−1.56e-1	6.06e-1	2.73e-1	3.32e-1	−1.21e-1
5	5.47e-1	2.42e-1	3.05e-1	−1.56e-1	−2.18e-1	−1.38e-1	−7.96e-2	−3.56e-1
6	−5.55e-1	−5.74e-1	1.95e-2	1.56e-2	−1.24e0	−9.13e-1	−3.25e-1	−1.52e-1
7	8.98e-1	6.84e-1	2.15e-1	1.72e-1	5.67e-1	5.19e-1	4.84e-2	8.65e-2
8	1.15e0	8.09e-1	3.40e-1	6.72e-1	9.20e-1	6.96e-1	2.25e-1	6.16e-1

the node 4 as the most important, as expected. In case that all nodes have the same absorption rate equal to 1, except for node 4 that has larger absorption rate, we observe that the $L^d \mathbf{1}$ measure, taking into account the distance from the highly absorbing node, ranks the nodes 1, 3, 5, and 7 as the most important, whereas the measure based on $L^d W \mathbf{1}$ is more "flexible", being adjusted depending on how large is the absorption rate of node 4.

Let us consider the unbalanced, unweighted graph depicted in Fig. 3 (right). In Tables 4, 5, 6 and 7 we present the centrality scores of its nodes using the $L^d W_s \mathbf{1}$, $L^d W_o \mathbf{1}$, $L^d W_i \mathbf{1}$ and $L^d \mathbf{1}$ centrality measures. In Case 1 we consider absorption rates $\mathbf{d} = \mathbf{1}$, whereas in Cases 2 and 3 we consider $d_i = 1$, $i = 1, \ldots, 8,\ i \neq 4$, $d_4 > 1$, in Cases 4

Table 7
Centrality scores of the graph of Fig. 3 (right) with absorption rates $d_i = 1$, $i = 1, \ldots, 7$ and $d_8 = 1.5$ (Case 6), $d_8 = 2$ (Case 7).

Node	Case 6				Case 7			
	$L^d W_s \mathbf{1}$	$L^d W_o \mathbf{1}$	$L^d W_i \mathbf{1}$	$L^d \mathbf{1}$	$L^d W_s \mathbf{1}$	$L^d W_o \mathbf{1}$	$L^d W_i \mathbf{1}$	$L^d \mathbf{1}$
1	−1.66e0	−8.77e-1	−7.80e-1	−2.57e-1	−9.06e-1	−5.16e-1	−3.91e-1	−6.25e-2
2	−4.47e-2	−7.08e-2	2.60e-2	−2.25e-1	5.94e-1	2.34e-1	3.59e-1	−6.25e-2
3	−1.66e0	−8.77e-1	−7.80e-1	−2.57e-1	−9.06e-1	−5.16e-1	−3.91e-1	−6.25e-2
4	7.94e-1	3.49e-1	4.45e-1	−1.28e-1	1.09e0	4.84e-1	6.09e-1	−6.25e-2
5	1.25e0	5.74e-1	6.71e-1	1.04e-3	1.09e0	4.84e-1	6.09e-1	−6.25e-2
6	−6.64e-1	−6.55e-1	−9.37e-3	−7.91e-2	−1.48e0	−1.09e0	−3.98e-1	−3.44e-1
7	1.22e0	8.36e-1	3.84e-1	2.42e-1	1.17e0	8.05e-1	3.67e-1	2.19e-1
8	5.10e-1	4.81e-1	2.91e-2	4.68e-1	−3.28e-1	5.47e-2	−3.83e-1	2.19e-1

and 5, $d_i = 1$, $i = 1, \ldots, 8$, $i \neq 5$, $d_5 > 1$, and in Cases 6 and 7, $d_i = 1$, $i = 1, \ldots, 7$, $d_8 > 1$. It should be mentioned that the centrality measure given by the row sums of L^d was proposed in [13] only for the case of balanced graphs. Here we include it only as a reference point and we do not make any claims as to its suitability for unbalanced graphs.

In Table 3 we notice that in the case of all equal absorptions, the $L^d W_s \mathbf{1}$ ranking is $5, 8, 7, 4, 6, 2, 1$ and 3, whereas for the case of indegrees ($L^d W_i \mathbf{1}$), the ranking is $5, 8, 6, 7, 4, 2, 1$ and 3 and for the case of outdegrees ($L^d W_o \mathbf{1}$), the ranking is $8, 7, 5, 4, 6, 2, 1$ and 3.

Setting $d_i = 1$, $i = 1, \ldots, 8$, $i \neq 4$ and $d_4 = 2$ (Table 5), the $L^d W_s \mathbf{1}$ ranking is $8, 5, 6, 7, 2, 4, 3, 1$, the $L^d W_i \mathbf{1}$ ranking is $6, 5, 8, 7, 2, 4, 3, 1$, and the $L^d W_o \mathbf{1}$ ranking is $8, 7, 5, 6, 2, 4, 3, 1$.

Considering $d_i = 1$, $i = 1, \ldots, 8$, $i \neq 5$ and $d_5 = 2$ (Table 6), the $L^d W_s \mathbf{1}$ ranking is $8, 2, 4, 7, 5, 1$ and $3, 6$, whereas the $L^d W_i \mathbf{1}$ ranking is $2, 4, 8, 7, 5, 1$ and $3, 6$, and the $L^d W_o \mathbf{1}$ ranking is $8, 7, 2, 4, 5, 1$ and $3, 6$.

If we set $d_i = 1$, $i = 1, \ldots, 8$, $i \neq 8$ and $d_8 = 2$ (Table 7), the $L^d W_s \mathbf{1}$ and the $L^d W_o \mathbf{1}$ ranking is $7, 4$ and $5, 2, 8, 1$ and $3, 6$, whereas for the case of the $L^d W_i \mathbf{1}$, the ranking is 4 and $5, 7, 2, 8, 1$ and $3, 6$.

From this example one can see again that the new centrality measures, while taking absorption into account, are also able to give connectivity more weight when determining node importance.

4. Numerical methods for the computation of the absorption inverse

From the above discussion we see how desirable is the efficient computation of the matrix L^d and of the action of L^d on a vector $\mathbf{b}$, $L^d \mathbf{b}$. In the sequel, we present a direct approach for the computation of L^d.

4.1. A direct method for the matrix L^d

Let L be a given $n \times n$ Laplacian matrix. We recall that L is a singular M-matrix [4]. We also recall (see [4]) that the LU factorization of an irreducible M-matrix, singular

or nonsingular, always exists. Moreover, it follows from results in [10] that no pivoting is necessary when computing the LU factorization of irreducible graph Laplacian matrices, since Gaussian elimination is stable for such matrices. Therefore, the following factorization always exists:

$$L = \mathcal{L}\,\mathcal{D}\,\mathcal{U},$$

where $\mathcal{L}$ and $\mathcal{U}$ are unit lower and upper triangular matrices, respectively, and $\mathcal{D}$ is a diagonal matrix of the form $\mathcal{D} = \mathrm{diag}(\delta_1, \delta_2, \ldots, \delta_{n-1}, 0)$, with $\delta_i > 0$ for $1 \leq i \leq n-1$. For undirected graphs, $L = L^T$ and $\mathcal{U} = \mathcal{L}^T$. We further note that any symmetric permutation PLP^T (with P a permutation matrix) enjoys the same properties as L.

Let us furthermore recall the following interesting property that holds for the matrix L [2]. Recall that L is irreducible since we assume that G is strongly connected.

Lemma 1. *Let L be a $n \times n$ Laplacian matrix. The matrix*

$$L^- = \mathcal{U}^{-1}\mathcal{D}^-\mathcal{L}^{-1}, \tag{5}$$

where $\mathcal{D}^- = \mathrm{diag}(\delta_1^{-1}, \delta_2^{-1}, \ldots, \delta_{n-1}^{-1}, 0)$, is a $\{1\text{-}2\}$ generalized inverse of L.

Lemma 2. *Suppose the Laplacian matrix L is partitioned as*

$$L = \begin{bmatrix} L_{n-1} & \mathbf{u} \\ \mathbf{w}^T & l_{n,n} \end{bmatrix},$$

where L_{n-1} is $(n-1) \times (n-1)$. Then,

$$L^- = \begin{bmatrix} L_{n-1}^{-1} & \mathbf{0}_{n-1,1} \\ \mathbf{0}_{1,n-1} & 0 \end{bmatrix}.$$

Proof. A straightforward calculation. □

Remark 2. The positive matrix L_{n-1}^{-1} is called the bottleneck matrix of L based at vertex n in [13, p. 129]. Note that L^- coincides with matrix M in [13, eqn. (20)].

Formula (2) [13, Theorem 3] forms the basis for a method to compute the matrix L^d. By considering as Y in (2) the generalized inverse L^- of formula (5) we have the following expression:

$$L^d = (I_n - VD)\mathcal{U}^{-1}\mathcal{D}^-\mathcal{L}^{-1}(I_n - DV)$$

which leads to Algorithm 1 below. In nearly all cases of practical interest, the Laplacian matrix L is sparse. Exploiting sparsity is crucial for the efficient implementation of

the proposed algorithms. When computing the triangular factorization $L = \mathcal{L}\mathcal{D}\mathcal{U}$, a fill-reducing ordering must be used. Hence, the matrix L is first permuted symmetrically and then factored. If P is the permutation matrix corresponding to the chosen ordering, then we compute the factorization $PLP^T = \mathcal{L}\mathcal{D}\mathcal{U}$. In case of a nonhomogeneous system $L\mathbf{x} = \mathbf{c}$, we solve the system $PLP^T\mathbf{y} = P\mathbf{c}$, then let $\mathbf{x} = P^T\mathbf{y}$. We have the following result.

Proposition 5. *The absorption inverse of the permuted Laplacian matrix is the permutation of the absorption inverse L^d that corresponds to the initial Laplacian matrix L, i.e. $(PLP^T)^d = PL^dP^T$, where P is the related permutation matrix.*

Proof. Let $\hat{D} = PDP^T$ and $\hat{V} = PV$. Note that $PL^dP^T = P(I_n - VD)L^-(I_n - DV)P^T$ and that $(PLP^T)^d = (I_n - \hat{V}\hat{D})PL^-P^T(I_n - \hat{D}\hat{V})$ since PL^-P^T is a {1-2} inverse of PLP^T, because $PL^-P^TPLP^TPL^-P^T = PL^-LL^-P^T = PL^-P^T$ and $PLP^TPL^-P^TPLP^T = PLL^-LP^T = PLP^T$. Additionally, $\hat{D}$ has as diagonal entries the absorption rates that correspond to the permuted matrix PLP^T. Also, for $\mathbf{v} \in \mathrm{Ker}(L)$ then $P\mathbf{v} \in \mathrm{Ker}(PLP^T)$ since $PLP^TP\mathbf{v} = PL\mathbf{v} = 0$. Therefore, $\hat{V} = P\mathbf{v}\mathbf{1}^T/\mathbf{d}^T\mathbf{v} = P\mathbf{v}\mathbf{1}^TP^T/\mathbf{d}^T\mathbf{v}$, since $\mathbf{1}^TP^T = \mathbf{1}^T$ and thus $\hat{V} = PVP^T$. Hence, we have $(PLP^T)^d = (I_n - PVP^TPDP^T)PL^-P^T(I_n - PDP^TPVP^T) = (PP^T - PVDP^T)PL^-P^T(PP^T - PDVP^T) = P(P^T - VDP^T)PL^-P^T(P - PDV)P^T = P(I_n - VD)L^-(I_n - DV)P^T = PL^dP^T$. □

In the sequel, the steps of the algorithm for computing the matrix L^d are presented.

Algorithm 1: Direct algorithm for L^d.

Input: $L \in \mathbb{R}^{n\times n}$ a Laplacian matrix, $\mathbf{d} \in \mathbb{R}^n$ a vector of absorption rates
Output: L^d
Obtain the triangular factorization $L = \mathcal{L}\mathcal{D}\mathcal{U}$
Find the normalized vector $\mathbf{v} \in \mathrm{Ker}(L)$
Compute the matrices
$L^- = \mathcal{U}^{-1}\mathcal{D}^-\mathcal{L}^{-1}$
$Y_1 = VDL^-$, $Y_2 = L^-DV$ and $Y_3 = Y_1DV$
Return $L^d = L^- - Y_1 - Y_2 + Y_3$

In the implementation of Algorithm 1 we make use of the colamd and symamd functions in MATLAB, for directed and undirected graphs respectively, as permutations. Also, in the computations of matrices Y_1 and Y_2 we make use of forward and backward substitution with $\mathcal{L}$ and $\mathcal{U}$.

Remark 3. If the graph is balanced, then the normalized vector $\mathbf{v} = (1/n)\mathbf{1} \in \mathrm{Ker}(L)$ since $L\mathbf{1} = 0$. Therefore, $\tilde{d} = \mathbf{d}^T\mathbf{v}$ is the mean value of the absorption rates and the matrix V has all its entries equal to $1/\sum_{i=1}^n d_i$.

Remark 4. In case of unbalanced graphs, when looking for a nonzero solution to $L\mathbf{v} = \mathbf{0}$, we factor L (or, in practice, a symmetric permutation PLP^T of it) as $L = \mathcal{L}\mathcal{D}\mathcal{U}$ and

solve the equivalent system $\mathcal{D}\mathcal{U}\mathbf{v} = \mathbf{0}$. Since the last equation of this system is the identity $\mathbf{0} = \mathbf{0}$, this is actually an underdetermined system of $n-1$ linear equations in n unknowns. Fixing the value of $v_n = 1$ yields the equivalent upper triangular system $\mathcal{U}\mathbf{v} = \mathbf{e}_n$, which is easily solved by backsubstitution. Since $\mathcal{U}^{-1}$ is a nonnegative matrix, so is $\mathbf{v}$. Normalization of $\mathbf{v}$ in the 1-norm produces the desired vector in $\mathrm{Ker}(L)$.

4.2. Computation of the vector $L^d\mathbf{b}$

Next, we show how to compute the action of L^d on a vector $\mathbf{b}$, i.e. $L^d\mathbf{b}$. The action of the absorption inverse on a vector $L^d\mathbf{b}$ is evaluated through the expression $(I_n - VD)Y(I_n - DV)\mathbf{b}$, by considering as Y the generalized inverse L^-, the pseudoinverse L^+ or the group inverse $L^\#$ of the matrix L. A crucial step in this computation is to solve the system $L\mathbf{x} = \mathbf{c}$ for $\mathbf{c} = \mathbf{b} - DV\mathbf{b}$. For this system, the following holds.

Proposition 6. *The system $L\mathbf{x} = \mathbf{c}$, for $\mathbf{c} = \mathbf{b} - DV\mathbf{b}$ is consistent.*

Proof. The system $L\mathbf{x} = \mathbf{c}$ for $\mathbf{c} = \mathbf{b} - DV\mathbf{b}$ is consistent since it holds $\mathbf{1}^T\mathbf{c} = \mathbf{1}^T(\mathbf{b} - DV\mathbf{b}) = \mathbf{1}^T\mathbf{b} - \mathbf{1}^TD\mathbf{v}\mathbf{1}^T\mathbf{b}/\tilde{d} = \mathbf{1}^T\mathbf{b} - \mathbf{1}^T\mathbf{b} = 0$, as $\mathbf{1}^TD = \mathbf{d}^T$ and $\mathbf{d}^T\mathbf{v} = \tilde{d}$. Hence $\mathbf{c} \perp \mathrm{Ker}(L^T)$ and therefore $\mathbf{c} \in \mathrm{Range}(L)$. □

i. A direct method

In this algorithm, the action of the absorption inverse on a vector $L^d\mathbf{b}$ is evaluated through the expression $(I_n - VD)L^-(I_n - DV)\mathbf{b} = (I_n - VD)L^-\mathbf{c}$, where $\mathbf{c} = \mathbf{b} - DV\mathbf{b}$, as follows.

Algorithm 2: Direct method for computing $L^d\mathbf{b}$.

Input: $L \in \mathbb{R}^{n\times n}$ a Laplacian matrix, $\mathbf{d} \in \mathbb{R}^n$ a vector of absorption rates, $\mathbf{b} \in \mathbb{R}^n$ a vector
Output: $L^d\mathbf{b}$
Obtain the triangular factorization $L = \mathcal{L}\mathcal{D}\mathcal{U}$
Find the normalized vector $\mathbf{v} \in \mathrm{Ker}(L)$
Compute the vectors
  $\mathbf{c} = \mathbf{b} - DV\mathbf{b}$ and
  $\mathbf{x} = L^-\mathbf{c}$
Return $L^d\mathbf{b} = \mathbf{x} - \mathbf{v}(\mathbf{d}^T\mathbf{x})/\tilde{d}$

Remark 5. In this Algorithm, the matrix L^- is not formed explicitly and the quantity $L^-\mathbf{c}$ is computed using the triangular factors of L. In particular, the procedure followed for solving $L\mathbf{v} = \mathbf{0}$ is described in Remark 4. In addition, when solving $L\mathbf{x} = \mathbf{c}$ with $\mathbf{c} \in \mathrm{Range}(L)$, $\mathbf{c} \neq \mathbf{0}$, we use again the triangular factors of L (or, in practice, of PLP^T). Formally, the solution is given by $\mathbf{x} = L^-\mathbf{c} = \mathcal{U}^{-1}(\mathcal{D}^-(\mathcal{L}^{-1}\mathbf{c}))$. Hence, computing x amounts to performing a forward substitution with $\mathcal{L}$, a diagonal scaling, and finally a back substitution with $\mathcal{U}$. Additionally, in the implementation of Algorithm 2 we make use of the colamd and symamd functions in MATLAB, for directed and undirected graphs respectively, as permutations.

ii. Iterative algorithms

In the sequel, the steps of the iterative algorithms are presented for undirected and directed graphs. In these algorithms, the action of the absorption inverse on a vector $L^d\mathbf{b}$ is evaluated through the expression $(I_n - VD)Y(I_n - DV)\mathbf{b}$, by considering as Y the pseudoinverse L^+ of the matrix L. The main computational task in these algorithms is to solve the system $L\mathbf{x} = \mathbf{c}$ for $\mathbf{c} = \mathbf{b} - DV\mathbf{b}$. Then, by considering the solution $\mathbf{x}$ of the system $L\mathbf{x} = \mathbf{c}$, it follows that $L^d\mathbf{b}$ is given by the expression $\mathbf{x} - \mathbf{v}(\mathbf{d}^T\mathbf{x})/\tilde{d}$.

Undirected case

We recall that for any symmetric positive semidefinite matrix A and $\mathbf{b} \in \text{Range}(A)$, $\mathbf{b} \neq \mathbf{0}$, the preconditioned conjugate gradients method with $\mathbf{x}_0 = \mathbf{0}$ converges to the pseudoinverse solution $A^+\mathbf{b}$ of $A\mathbf{x} = \mathbf{b}$ [14].

Algorithm 3: Iterative methods for computing $L^d\mathbf{b}$ for undirected case.

Input: $L \in \mathbb{R}^{n\times n}$ a Laplacian matrix, $\mathbf{d} \in \mathbb{R}^n$ a vector of absorption rates, $\mathbf{b} \in \mathbb{R}^n$ a vector
Output: $L^d\mathbf{b}$
Compute the normalized vector $\mathbf{v} = (1/n)\mathbf{1}$
Compute the vector $\mathbf{c} = \mathbf{b} - DV\mathbf{b}$
Solve the system $L\mathbf{x} = \mathbf{c}$ using the preconditioned conjugate gradient method
Return $L^d\mathbf{b} = \mathbf{x} - \mathbf{v}(\mathbf{d}^T\mathbf{x})/\tilde{d}$

As we will see in the section on numerical experiments, good results can be obtained using an incomplete LU (or incomplete LDL^T) factorization of the Laplacian as a preconditioner. It is known [5] that the no-fill incomplete LU factorization of a singular M-Matrix is well-defined. The effectiveness of this type of preconditioner can be improved if L is first permuted by a band-reducing ordering, such as a reverse Cuthill-McKee; see, e.g., [3]. In our implementation we make use of the symrcm function in MATLAB to reorder L prior to computing its incomplete LU factorization.

Directed case

Here, we consider the use of an iterative method for computing the action of L^d on a vector $\mathbf{b}$ for the directed case. In the procedure given in Algorithm 4 below, the two linear systems are solved using the preconditioned GMRES method [20].

Algorithm 4: Iterative methods for computing $L^d\mathbf{b}$ for directed case.

Input: $L \in \mathbb{R}^{n\times n}$ a Laplacian matrix, $\mathbf{d} \in \mathbb{R}^n$ a vector of absorption rates, $\mathbf{b} \in \mathbb{R}^n$ a vector
Output: $L^d\mathbf{b}$
Solve the system $L\mathbf{v} = \mathbf{0}$
Compute the vector $\mathbf{c} = \mathbf{b} - DV\mathbf{b}$
Solve the system $L\mathbf{x} = \mathbf{c}$
Return $L^d\mathbf{b} = \mathbf{x} - \mathbf{v}(\mathbf{d}^T\mathbf{x})/\tilde{d}$

The convergence properties of GMRES applied to singular systems have been studied in many papers; see, e.g., the recent report [18]. Laplacian matrices of graphs satisfy the

condition $\mathrm{Ker}(L) \cap \mathrm{Range}(L) = \{\mathbf{0}\}$. For such matrices it is known that GMRES applied to a consistent system $L\mathbf{x} = \mathbf{c}$ starting from an arbitrary initial guess $\mathbf{x}_0 \in \mathbb{R}^n$ converges to a solution of $L\mathbf{x} = \mathbf{c}$ without breakdowns. Moreover, the computed solution is of the form $\mathbf{x} = L^{\#}\mathbf{c} + (I_n - L^{\#}L)\mathbf{x}_0$. Hence, for $\mathbf{x}_0 = \mathbf{0}$ we obtain $\mathbf{x} = L^{\#}\mathbf{c}$. Since the group inverse is a particular case of $\{1\}$-inverse, this solution satisfies our requirements. It is easy to see that the same holds when preconditioning is used.

5. Updating/downdating the graph

Let us suppose that we have an updating of the graph, such as the addition of an edge between two nodes i and j with $i \neq j$. This can be interpreted as a rank-one change in the adjacency matrix A of the form $A_1 = A + \mathbf{e}_i\mathbf{e}_j^T$ in case of directed graphs or a rank-two change in the matrix A of the form $A_1 = A + (\mathbf{e}_i + \mathbf{e}_j)(\mathbf{e}_i + \mathbf{e}_j)^T$ in case of undirected graphs. Assume that we have a rank-one change in the matrix A. In this case, the diagonal matrix W is changed into $W_1 = W + \mathbf{e}_j\mathbf{e}_j^T$ and therefore, the Laplacian matrix of A_1, denoted as L_1, has the form $L_1 = W_1 - A_1 = L - (\mathbf{e}_i - \mathbf{e}_j)\mathbf{e}_j^T$. Then we have the following results. The case of a deletion of an edge, i.e. downdating of the graph, is treated along similar lines.

Proposition 7. *The generalized inverse* (5) *of the Laplacian matrix* $L_1 = L - (\mathbf{e}_i - \mathbf{e}_j)\mathbf{e}_j^T$, *denoted as* L_1^-, *is* $L_1^- = L^- - \dfrac{1}{q_j}\mathbf{q}L^-_{j,:}$, *where* $L^-_{j,:}$ *denotes the* j*th row of the matrix* L^- *and* $\mathbf{q} = \dfrac{1}{v_n}\mathbf{v}$. *In the case of balanced graphs, the aforementioned expression of* L_1^- *can be written as* $L_1^- = L^- - \mathbf{1}L^-_{j,:}$.

Proof. Let us consider the formula $L_1^- = (I_n - M)L^-$, given in [9, page 32], where $M = \dfrac{1}{\mathbf{e}_j^T FF^T \mathbf{e}_j} FF^T\mathbf{e}_j\mathbf{e}_j^T$ and $F = I_n - L^-L$. It follows from Lemma 2 that the matrix F has all its entries zero except for its last column, which is equal to $\begin{bmatrix} -L_{n-1}^{-1}\mathbf{u} \\ 1 \end{bmatrix}$. In turn, this is equal to the vector $\mathbf{q} = \dfrac{1}{v_n}\mathbf{v}$. To see this, recall that $\mathbf{v}$ is the unique positive vector in the null space of L with entries adding up to 1. Recall the partitioning of L in Lemma 2. Our claim is that

$$\frac{1}{v_n}\mathbf{v} = \begin{bmatrix} -L_{n-1}\mathbf{u} \\ 1 \end{bmatrix}.$$

Note that the vector on the right hand side is positive, since the entries of L_{n-1}^{-1} are strictly positive and $\mathbf{u}$ is a nonpositive, nonzero vector. We have

$$\begin{bmatrix} L_{n-1} & \mathbf{u} \\ \mathbf{w}^T & l_{n,n} \end{bmatrix} \begin{bmatrix} -L_{n-1}^{-1}\mathbf{u} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{n-1,1} \\ l_{n,n} - \mathbf{w}^T L_{n-1}^{-1}\mathbf{u} \end{bmatrix}.$$

From the (unique) block LU factorization of L,

$$L = \begin{bmatrix} L_{n-1} & \mathbf{u} \\ \mathbf{w}^T & l_{n,n} \end{bmatrix} = \begin{bmatrix} I_{n-1} & \mathbf{0}_{n-1,1} \\ \mathbf{w}^T L_{n-1}^{-1} & 1 \end{bmatrix} \begin{bmatrix} L_{n-1} & \mathbf{u} \\ \mathbf{0}_{1,n-1} & 0 \end{bmatrix}$$

we obtain $l_{n,n} = \mathbf{w}^T L_{n-1}^{-1}\mathbf{u}$. This shows that the last column of F is in the null space of L and therefore (up to the normalization factor v_n) it must equal $\mathbf{v}$. Then, the matrix M has only one nonzero column, namely its jth column, which is equal to the vector $\frac{1}{q_j}\mathbf{q}$. Hence, $ML^- = \frac{1}{q_j}\mathbf{q}L^-_{j,:}$. In the case of balanced graphs, $\mathbf{q} = \mathbf{1}$. □

Proposition 8. *The absorption inverse of L_1, L_1^d, can be computed by updating L^d by the formula*

$$L_1^d = L^d - (I_n - VD)L^-DK - KD(I_n - M)L^-(I_n - D(V + K)),$$

where $M = \frac{1}{q_j}\mathbf{q}\mathbf{e}_j^T$ *and* $\mathbf{q} = \frac{1}{v_n}\mathbf{v}$, $K = \frac{1}{\mathbf{d}^T\mathbf{v} - k_1}(Vk_1 - K_2)$, $k_1 = \mathbf{d}^T\mathbf{z}$, $K_2 = \mathbf{z}\mathbf{1}^T$, $\mathbf{z} = \frac{v_j}{L^-_{ji} - L^-_{jj} - 1}(L^-_{:,i} - L^-_{:,j})$, *$v_j$ is the jth entry of $\mathbf{v}$ and $L^-_{:,j}$ denotes the jth column of the matrix L^-.*

Proof. The vector $\tilde{\mathbf{v}} \in \mathrm{Ker}(L_1)$ can be computed by the formula $\tilde{\mathbf{v}} = \mathbf{v} - \frac{v_j}{\mathbf{e}_j^T L^-(\mathbf{e}_i - \mathbf{e}_j) - 1}(L^-(\mathbf{e}_i - \mathbf{e}_j))$ [9, equation (1.4)]. Then, $\tilde{V} = \frac{1}{\mathbf{d}^T\tilde{\mathbf{v}}}\tilde{\mathbf{v}}\mathbf{1}^T = V + K$, where $K = \frac{1}{\mathbf{d}^T\mathbf{v} - k_1}(Vk_1 - K_2)$, $k_1 = \mathbf{d}^T\mathbf{z}$, $K_2 = \mathbf{z}\mathbf{1}^T$, $\mathbf{z} = \frac{v_j}{\mathbf{e}_j L^-(\mathbf{e}_i - \mathbf{e}_j) - 1}(L^-(\mathbf{e}_i - \mathbf{e}_j))$ and v_j is the jth entry of v. Therefore, $L_1^d = (I_n - \tilde{V}D)L_1^-(I_n - D\tilde{V}) = L^d - (I_n - VD)L^-DK - KD(I_n - M)L^-(I_n - D(V + K))$, since $(I_n - VD)M = \mathbf{0}_{n,n}$. □

It can be seen that the absorption inverse of the modified graph is at most a rank-two modification of the absorption inverse of the original graph.

6. Estimation of individual entries of the absorption inverse matrix for undirected graphs

In some situations, individual entries of L^d are of interest. It is desirable to be able to compute or estimate these entries without having to compute the whole matrix L^d. For estimating specific entries of the matrix L^d we can consider estimates of the bilinear form $\mathbf{y}^T L^d\mathbf{x}$, $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ for the specific selection of the vectors $\mathbf{y}$, $\mathbf{x}$ to be the ith and the jth columns of the identity matrix, respectively. In particular, in the case of an undirected graph, considering formula $L^d = (I_n - VD)L^+(I_n - DV)$ and $\mathbf{x} = \mathbf{y} = \mathbf{e}_i$ we have

$$L^d_{ii} = \mathbf{e}_i^T L^d\mathbf{e}_i = (\mathbf{e}_i^T - \mathbf{e}_i^T VD)L^+(\mathbf{e}_i - DV\mathbf{e}_i) = \mathbf{u}^T L^+\mathbf{u},$$

where $\mathbf{u} = \mathbf{e}_i - DV\mathbf{e}_i$ and $L^+ = \left(L + \frac{1}{n}\mathbf{1}\mathbf{1}^T\right)^{-1} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$ [24]. Hence, we have

$$L^d_{ii} = \mathbf{u}^T \left(L + \frac{1}{n}\mathbf{1}\mathbf{1}^T\right)^{-1} \mathbf{u} - \frac{1}{n}(\mathbf{u}^T\mathbf{1})^2.$$

Proposition 9. *For an undirected graph, it holds* $L^d_{ii} = \mathbf{u}^T \left(L + \frac{1}{n}\mathbf{1}\mathbf{1}^T\right)^{-1} \mathbf{u}$, *where* $\mathbf{u} = \mathbf{e}_i - DV\mathbf{e}_i$.

Proof. It holds $L^d_{ii} = \mathbf{u}^T \left(L + \frac{1}{n}\mathbf{1}\mathbf{1}^T\right)^{-1} \mathbf{u} - \frac{1}{n}(\mathbf{u}^T\mathbf{1})^2$, where $\mathbf{u} = \mathbf{e}_i - DV\mathbf{e}_i$ and $\frac{1}{n}(\mathbf{u}^T\mathbf{1})^2 = 0$ since $\mathbf{u}^T\mathbf{1} = \mathbf{e}_i^T\mathbf{1} - \mathbf{e}_i^T VD\mathbf{1} = 1 - \mathbf{e}_i^T v\mathbf{1}^T\mathbf{d}/\tilde{d}$ as $D\mathbf{1} = \mathbf{d}$. Also, we have $\mathbf{1}^T\mathbf{d} = \sum_{i=1}^n d_i$, $\tilde{d} = \mathbf{d}^T\mathbf{v} = \sum_{i=1}^n d_i/n$ and $\mathbf{v} = (1/n)\mathbf{1}$. Therefore, $\mathbf{u}^T\mathbf{1} = 1 - \mathbf{e}_i^T\mathbf{v}\mathbf{1}^T\mathbf{d}/\tilde{d} = 1 - \mathbf{e}_i^T\mathbf{1} = 1 - 1 = 0$. □

We can obtain approximations and upper/lower bounds for the quantity $\mathbf{u}^T \left(L + \frac{1}{n}\mathbf{1}\mathbf{1}^T\right)^{-1} \mathbf{u}$ through an approach based on Gauss quadrature rules and the Lanczos algorithm [11]. Let $B = L + \frac{1}{n}\mathbf{1}\mathbf{1}^T$. Note that B is symmetric positive definite. Consider the spectral decompositions

$$B = Q\Lambda Q^T, \quad B^{-1} = Q\Lambda^{-1}Q^T,$$

where Q is orthogonal and Λ diagonal, with the eigenvalues of B in nondecreasing order down the main diagonal. For $\mathbf{u} \in \mathbb{R}^n$ we have

$$\mathbf{u}^T B^{-1}\mathbf{u} = \mathbf{u}^T Q\Lambda^{-1}Q^T\mathbf{u} = \mathbf{p}^T\Lambda^{-1}\mathbf{p} = \sum_{i=1}^n \lambda_i^{-1}p_i^2,$$

where $\mathbf{p} = Q^T\mathbf{u}$. Rewrite this as a Riemann-Stieltjes integral:

$$\mathbf{u}^T B^{-1}\mathbf{u} = \int_a^b \lambda^{-1} d\mu(\lambda),$$

where the measure $d\mu$ is defined via

$$\mu(\lambda) = \begin{cases} 0 & \lambda < a = \lambda_1 \\ \sum_{j=1}^i p_j^2 & \lambda_i \le \lambda < \lambda_{i+1} \\ \sum_{j=1}^N p_j^2 & b = \lambda_n \le \lambda. \end{cases}$$

Evaluation of the Riemann-Stieltjes integral requires knowledge of the eigendecomposition of B, which is not available in general. The integral, however, can be approximated by means of Gaussian quadrature rules [1,11]. Recall that the general Gauss-type quadrature rule is

$$\int_a^b f(\lambda)d\mu(\lambda) = \sum_{j=1}^{m} w_j f(t_j) + \sum_{k=1}^{M} v_k f(z_k) + R[f],$$

where the nodes t_j are the zeros of orthogonal polynomials (with respect to the measure $d\mu$), the nodes z_j (if any) are prescribed, and the weights w_j and v_k are to be determined. The error term $R[f]$ is given by

$$R[f] = \frac{f^{(2m+M)}(\eta)}{(2m+M)!} \int_a^b \prod_{k=1}^{M}(\lambda - z_k) \left[\prod_{j=1}^{m}(\lambda - t_j)\right]^2 d\mu(\lambda),$$

where $\eta \in (a,b)$, assuming that the function f is of class $C^{(2m+M)}$ on $[a,b]$.

More precisely, we have three types of Gaussian quadrature rules:

- Gauss: $M = 0$,
- Gauss–Radau: $M = 1$, $z_1 = a$ or $z_1 = b$,
- Gauss–Lobatto: $M = 2$, $z_1 = a$ and $z_2 = b$.

The Lanczos algorithm (with starting vector $\mathbf{u}$) can be used to generate the family of (discrete) orthogonal polynomials with respect to $d\mu$. These polynomials satisfy a three-term recurrence, the coefficients of which are the entries of the tridiagonal matrix J_m generated by the Lanczos algorithm after m steps:

$$J_m = \begin{bmatrix} \omega_1 & \gamma_1 & & & & \\ \gamma_1 & \omega_2 & \gamma_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \gamma_{m-2} & \omega_{m-1} & \gamma_{m-1} & \\ & & & \gamma_{m-1} & \omega_m & \end{bmatrix}.$$

The eigenvalues of J_m are the Gauss quadrature nodes, whereas the Gauss weights are given by the squares of the first entries of the normalized eigenvectors of J_m. Alternatively, the Gauss quadrature rule can also be computed as the $(1,1)$ entry of J_m^{-1}, which can be evaluated incrementally for $m = 1, 2, \ldots$ (see [11, Section 6.2.4]). When $\mathbf{u} = \mathbf{e}_i - DV\mathbf{e}_i$, we obtain approximations of the (i,i) entry of L^d. From the expression of the quadrature error term given above and the fact that $M = 0$ for the Gauss rule, we see that the error is always positive and therefore the Gauss approximation to the (i,i) entry of L^d is a lower bound. The Gauss–Radau rule and the Gauss–Lobatto rules can be

used to obtain upper bounds; all these bounds become increasingly tight as m increases. Taking the mean of these lower and upper bounds results in estimates for the quantities of interest, together with bounds on the corresponding error. Usually, a few Lanczos steps (i.e., a few quadrature nodes) are sufficient to obtain very good approximations.

The non-diagonal entries of L^d can be approximated by using the polarization identity $\mathbf{z}^T B^{-1}\mathbf{y} = \frac{1}{4}((\mathbf{s}, B^{-1}\mathbf{s}) - (\mathbf{t}, B^{-1}\mathbf{t}))$, for $\mathbf{z}, \mathbf{y} \in \mathbb{R}^n$, $\mathbf{s} = \mathbf{z} + \mathbf{y}$, $\mathbf{t} = \mathbf{z} - \mathbf{y}$. Therefore, for $\mathbf{z} = \mathbf{e}_i$ and $\mathbf{y} = \mathbf{e}_j$ we have

$$L^d_{ij} = \frac{1}{4}\left((\mathbf{s}, B^{-1}\mathbf{s}) - (\mathbf{t}, B^{-1}\mathbf{t})\right) + \frac{1}{4n}\left((\mathbf{t}^T\mathbf{1})^2 - (\mathbf{s}^T\mathbf{1})^2\right),$$
$$\mathbf{s} = \mathbf{e}_i + \mathbf{e}_j - DV(\mathbf{e}_i + \mathbf{e}_j),\ \mathbf{t} = \mathbf{e}_i - \mathbf{e}_j - DV(\mathbf{e}_i - \mathbf{e}_j).$$

Proposition 10. *For an undirected graph, it holds* $L^d_{ij} = \frac{1}{4}\left((\mathbf{s}, B^{-1}\mathbf{s}) - (\mathbf{t}, B^{-1}\mathbf{t})\right)$, $\mathbf{s} = \mathbf{e}_i + \mathbf{e}_j - DV(\mathbf{e}_i + \mathbf{e}_j)$, $\mathbf{t} = \mathbf{e}_i - \mathbf{e}_j - DV(\mathbf{e}_i - \mathbf{e}_j)$.

Proof. It holds $L^d_{ij} = \frac{1}{4}\left((\mathbf{s}, B^{-1}\mathbf{s}) - (\mathbf{t}, B^{-1}\mathbf{t})\right) + \frac{1}{4n}\left((\mathbf{t}^T\mathbf{1})^2 - (\mathbf{s}^T\mathbf{1})^2\right)$, $\mathbf{s} = \mathbf{e}_i + \mathbf{e}_j - DV(\mathbf{e}_i + \mathbf{e}_j)$, $\mathbf{t} = \mathbf{e}_i - \mathbf{e}_j - DV(\mathbf{e}_i - \mathbf{e}_j)$ and $\frac{1}{4n}\left((\mathbf{t}^T\mathbf{1})^2 - (\mathbf{s}^T\mathbf{1})^2\right) = 0$, since $\mathbf{t}^T\mathbf{1} = \mathbf{s}^T\mathbf{1} = 0$ along the same lines as in the proof of Proposition 9. □

Then, estimates for the elements L^d_{ij} can be obtained by approximating the quadratic forms on the right hand side of the identity.

7. Numerical examples

In this section, numerical experiments are presented for the computation of the matrix vector product $L^d\mathbf{b}$, where $\mathbf{b}$ is a vector with all ones, through the iterative methods (Example I) and through the direct method (Example II). In cases that the given graph is not strongly connected, i.e. $\text{rank}(L) < n - 1$, we consider the largest connected graph component (LCC). In Example II the matrix L^d is also computed through the direct approach. Additionally, individual entries of the absorption inverse matrix are estimated in Example III following the procedure described in section 6. In these experiments, random absorption rates are considered.

The networks tested in the following Tables are obtained by the SuiteSparse matrix collection [21]. The computations are performed in MATLAB R2015b 64-bit (win64), on an Intel Core i5-6200U with 8 GB RAM at 2.3 GHz.

Example I. Computation of $L^d\mathbf{b}$ through the iterative algorithms for directed and undirected graphs. We compute $L^d\mathbf{b}$ for directed (Tables 8-9) and undirected graphs (Tables 10-13) of fairly large dimension varying from $\mathcal{O}(10^4) - \mathcal{O}(10^6)$ by employing iterative methods with specific preconditioners and with a tolerance of $\mathcal{O}(10^{-6})$. In particular, we used the Jacobi (or diagonal) preconditioner, the symmetric Gauss-Seidel

(SGS) preconditioner, the ILU preconditioner with no-fill and the block ILU preconditioner with number of blocks 8, 16 and 32. Here, by block ILU preconditioner we mean a block Jacobi type preconditioner with a no-fill ILU decomposition of each diagonal block replacing the exact blocks. Whenever n is not divisible by one of these integers, the last block will have smaller size than the remaining ones.

Tables 8-13 summarize the attained results as follows. The first columns of the Tables 8, 10, 12 display the characteristics of the networks, i.e. the name, the number of nodes n of the network and of the LCC if the network is not strongly connected, as well as the number of edges of the network or its LCC. In the subsequent columns of these Tables and in Tables 9, 11, 13 we report the number of iterations required for attaining the given tolerance and the execution time (in seconds) for each applied preconditioner. Also, for undirected graphs an algebraic multigrid (AMG) solver of the symmetric linear system $L\mathbf{x} = \mathbf{c}$ is employed with a tolerance of $\mathcal{O}(10^{-6})$ and maximum number of iterations equal to 2000 [12], [17]. The results are presented in the last two columns of Tables 11 and 13 and concern the standard V-cycle AMG solver, using the Heavy Edge Coarsening (HEC) algorithm [23].

All the presented results are averaged and rounded (for the iteration counts) over 5 runs, since we are using random absorption rates. For the results presented in Table 8, the GMRES method is restarted every 100 iterations. In this Table the iterations required are reported in brackets $[\cdot, \cdot]$, where the first number corresponds to the iterations required for the computation of $\mathbf{v} \in \mathrm{Ker}(L)$ and the second number corresponds to the iterations required for the solution of the system $L\mathbf{x} = \mathbf{c}$.

In Tables 8-13 we notice that the quantity $L^d\mathbf{b}$ can be computed efficiently in seconds or at most a few minutes of CPU time even for large graphs with up to 36 million edges. In particular, concerning the directed graphs, the results presented in Tables 8 and 9 show that the ILU preconditioner attains the fastest convergence and execution time in most of the test cases. Although the block ILU preconditioners are not as efficient as the ILU preconditioner, their performance is presented since they can be easily used in a parallel implementation reducing in this way the execution time of the whole computation. Also, we observe that for the web-Google network, the method does not converge using the Jacobi preconditioner (Table 8), considering the maximum number of iterations equal to 50000.

Concerning the undirected graphs, the results presented in Tables 10-13 show again that the ILU preconditioner results in the fastest convergence and execution time in the most of the graphs that are tested, whereas the Jacobi preconditioner requires the most iterations and execution time. In these Tables, the results of the PCG method using various preconditioners are also compared with those using an algebraic multigrid (AMG) solver of the symmetric linear system $L\mathbf{x} = \mathbf{c}$. In these experiments we notice that AMG is generally much slower than the incomplete factorization preconditioned CG and in several cases it failed to convergence in 2000 iterations.

Table 8
Execution time in seconds and number of iterations for computing $L^d\mathbf{b}$, where $\mathbf{b}$ is a vector with all ones (directed case). The symbol + stands for failure to converge.

Network	n	Nodes of LCC	Edges of LCC	Jacobi		SGS	
				Its	Time	Its	Time
ca-HepPh	12008	11204	235268	[57, 86]	3.855e-1	[32, 56]	2.662e-1
email-Enron	36692	33696	361622	[46, 83]	7.312e-1	[34, 52]	5.739e-1
p2p-Gnutella31	62586	14149	50916	[32, 32]	1.115e-1	[17, 17]	7.190e-2
enron	69244	8271	147353	[45, 43]	1.392e-1	[28, 28]	1.141e-1
soc-Epinions1	75888	32223	443506	[39, 80]	6.577e-1	[24, 49]	4.913e-1
soc-Slashdot0811	77360	70355	888662	[34, 42]	6.918e-1	[22, 26]	6.083e-1
Wordnet3	82670	13755	37497	[98, 132]	6.704e-1	[54, 68]	3.560e-1
internet	124651	6437	18327	[444, 473]	1.689e0	[201, 198]	8.873e-1
Stanford	281903	150532	1576314	[202, 518]	4.179e1	[201, 308]	3.206e1
Linux_call_graph	324085	2760	6425	[51, 50]	5.470e-2	[26, 25]	3.354e-2
cnr-2000	325557	112023	1646332	[401, 1281]	3.995e1	[202, 1237]	5.294e1
NotreDame_www	325729	34643	179725	[303, 679]	6.638e0	[141, 334]	4.408e0
web-NotreDame	325729	53968	304685	[401, 963]	1.339e1	[195, 512]	8.478e0
Stanford_Berkeley	683446	333752	4509784	[141, 248]	5.729e1	[101, 691]	1.386e2
web-BerkStan	685230	334857	4523232	[540, 1405]	2.813e2	[197, 428]	1.031e2
flickr	820878	527476	9357071	[164, 241]	9.422e1	[82, 117]	5.417e1
eu-2005	862664	752725	17933415	[224, 963]	4.170e2	[106, 715]	3.542e2
web-Google	916428	434818	3419124	+	+	[665, 4040]	1.029e3
in-2004	1382908	593687	7827263	[674, 2430]	8.220e2	[541, 1543]	6.376e2
wikipedia-20051105	1634989	1103453	18245140	[39, 41]	2.777e1	[23, 25]	2.255e1
wiki-Talk	2394385	111881	1477893	[26, 28]	7.832e-1	[15, 17]	7.814e-1
wikipedia-20061104	3148440	2104115	36125805	[39, 36]	5.297e1	[22, 22]	4.272e1

Table 9
Execution time in seconds and number of iterations for computing $L^d\mathbf{b}$, where $\mathbf{b}$ is a vector with all ones (directed case).

Network	ILU		Block - ILU-8		Block - ILU-16		Block - ILU-32	
	Its	Time	Its	Time	Its	Time	Its	Time
ca-HepPh	[12, 15]	1.503e-1	[30, 38]	2.306e-1	[32, 41]	1.939e-1	[31, 42]	2.068e-1
email-Enron	[14, 19]	2.176e-1	[38, 59]	6.015e-1	[40, 67]	6.585e-1	[40, 61]	6.102e-1
p2p-Gnutella31	[12, 12]	4.775e-2	[29, 29]	1.042e-1	[30, 31]	1.144e-1	[31, 31]	1.196e-1
enron	[12, 12]	5.153e-2	[31, 31]	1.049e-1	[33, 33]	1.134e-1	[34, 35]	1.142e-1
soc-Epinions1	[13, 22]	2.432e-1	[34, 63]	6.123e-1	[33, 63]	5.681e-1	[33, 64]	5.495e-1
soc-Slashdot0811	[11, 15]	3.709e-1	[25, 29]	5.881e-1	[30, 36]	6.948e-1	[32, 37]	6.876e-1
Wordnet3	[17, 18]	6.452e-2	[79, 93]	5.406e-1	[82, 98]	5.328e-1	[86, 101]	5.831e-1
internet	[100, 101]	4.943e-1	[359, 295]	1.340e0	[397, 339]	1.356e0	[428, 407]	1.545e0
Stanford	[89, 561]	4.142e1	[201, 513]	4.658e1	[182, 528]	4.338e1	[202, 656]	4.530e1
Linux_call_graph	[17, 17]	1.560e-2	[42, 41]	4.458e-2	[45, 44]	4.818e-2	[46, 45]	5.047e-2
cnr-2000	[101, 502]	1.762e1	[142, 1672]	5.011e1	[104, 2733]	6.973e1	[184, 2714]	7.043e1
NotreDame_www	[88, 255]	2.560e0	[195, 819]	7.450e0	[200, 1142]	9.335e0	[201, 1396]	1.123e1
web-NotreDame	[91, 2303]	2.696e1	[201, 1340]	1.632e1	[219, 1461]	1.846e1	[220, 2225]	2.639e1
Stanford_Berkeley	[101, 391]	7.962e1	[101, 411]	8.213e1	[102, 607]	1.067e2	[102, 606]	1.053e2
web-BerkStan	[127, 415]	8.632e1	[201, 381]	9.558e1	[225, 1402]	2.469e2	[301, 973]	1.921e2
flickr	[35, 58]	2.183e1	[101, 201]	8.370e1	[101, 201]	8.101e1	[101, 201]	8.015e1
eu-2005	[81, 249]	1.313e2	[191, 9062]	4.239e2	[192, 765]	3.628e2	[132, 1000]	4.269e2
web-Google	[304, 781]	2.291e2	[601, 1818]	5.081e2	[400, 4039]	9.464e2	[500, 4371]	1.018e3
in-2004	[269, 1635]	5.814e2	[624, 1776]	6.880e2	[319, 947]	3.487e2	[546, 1468]	5.551e2
wikipedia-20051105	[12, 12]	1.074e1	[29, 31]	2.151e1	[31, 35]	2.254e1	[34, 36]	2.393e1
wiki-Talk	[10, 10]	6.067e-1	[23, 24]	9.343e-1	[23, 24]	8.633e-1	[23, 24]	7.806e-1
wikipedia-20061104	[14, 12]	2.637e1	[31, 30]	4.771e1	[35, 33]	5.025e1	[36, 34]	5.031e1

Table 10
Execution time in seconds and number of iterations for computing $L^d\mathbf{b}$, where $\mathbf{b}$ is a vector with all ones (undirected case).

Network	n	Edges	Jacobi		SGS		ILU	
			Its	Time	Its	Time	Its	Time
cs4	22499	87716	220	1.645e-1	86	1.299e-1	77	1.260e-1
as-22July06	22963	96872	62	5.681e-2	30	5.288e-2	22	4.149e-2
fe_tooth	78136	905182	242	9.784e-1	108	9.916e-1	84	8.058e-1
fe_rotor	99617	1324862	278	1.496e0	123	1.543e0	101	1.289e0
fe_ocean	143437	819186	599	4.214e0	212	2.568e0	206	2.478e0
coAuthorsCiteseer	227320	1628268	305	4.107e0	124	3.480e0	48	1.410e0
citationCiteseer	268495	2313294	243	4.224e0	101	3.665e0	79	2.959e0
coAuthorsDBLP	299067	1955352	173	3.163e0	70	2.635e0	34	1.403e0
auto	448695	6629222	433	1.359e1	191	1.285e1	154	1.058e1
coPapersDBLP	540486	30491458	224	1.917e1	93	1.842e1	40	1.076e1
tx2010	914231	4456272	2616	1.304e2	1061	9.956e1	784	7.340e1
NACA0015	1039183	6229636	3712	2.026e2	1529	1.566e2	1132	1.108e2
belgium_osm	1441295	3099940	14722	9.207e2	5462	5.335e2	2613	2.573e2
netherlands_osm	2216688	4882476	25219	2.576e3	9521	1.466e3	5108	7.835e2
M6	3501776	21003872	6261	1.129e3	2478	7.961e2	1827	6.130e2
333SP	3712815	22217266	8876	1.663e3	3599	1.200e3	2653	9.055e2
venturiLevel3	4026819	16108474	10079	1.804e3	3623	1.023e3	3282	9.626e2

Table 11
Execution time in seconds and number of iterations for computing $L^d\mathbf{b}$, where $\mathbf{b}$ is a vector with all ones (undirected case). The symbol † stands for failure to converge due to instability.

Network	Block - ILU-8		Block - ILU-16		Block - ILU-32		AMG	
	Its	Time	Its	Time	Its	Time	Its	Time
cs4	103	1.598e-1	122	2.111e-1	162	2.435e-1	121	1.539e0
as-22July06	58	7.939e-2	59	7.221e-2	59	7.334e-2	49	3.268e-1
fe_tooth	161	1.399e0	187	1.497e0	204	1.618e0	285	7.608e0
fe_rotor	159	1.931e0	189	2.083e0	202	2.243e0	+	+
fe_ocean	253	3.038e0	278	3.173e0	316	3.548e0	281	1.106e1
coAuthorsCiteseer	106	2.777e0	108	2.698e0	113	2.828e0	+	+
citationCiteseer	104	3.301e0	117	3.354e0	150	4.175e0	454	4.398e1
coAuthorsDBLP	79	2.972e0	91	2.916e0	96	3.092e0	517	5.094e1
auto	219	1.461e1	252	1.617e1	318	1.959e1	+	+
coPapersDBLP	74	1.340e1	100	1.663e1	105	1.686e1	888	4.149e2
tx2010	902	8.484e1	976	8.914e1	1093	1.005e2	†	†
NACA0015	1258	1.234e2	1293	1.242e2	1362	1.317e2	1294	3.567e2
belgium_osm	2851	2.765e2	3050	2.918e2	3526	3.383e2	+	+
netherland_osm	5442	8.230e2	5762	8.672e2	6067	9.202e2	+	+
M6	2114	6.774e2	2101	6.907e2	2189	7.038e2	+	+
333SP	3228	1.071e3	3524	1.261e3	3738	1.240e3	+	+
venturiLevel3	3558	1.026e3	3567	1.006e3	3593	1.009e3	+	+

Table 12
Execution time in seconds and number of iterations for computing $L^d\mathbf{b}$, where $\mathbf{b}$ is a vector with all ones (undirected case).

Network	n	Nodes of LCC	Edges of LCC	Jacobi		SGS	
				Its	Time	Its	Time
roadNet-CA	1971281	1957027	5520776	8342	7.762e2	3266	5.201e2
roadNet-PA	1090920	1087562	3083028	5665	2.985e2	2098	1.890e2
roadNet-TX	1393383	1351137	3758402	10301	6.601e2	4088	4.492e2
as-Skitter	1696415	1694616	22188418	357	5.575e1	167	5.099e1
hollywood-2009	1139905	1069126	113682432	157	6.429e1	68	7.071e1
packing-500x100x100-b050	2145852	2145839	34976486	1062	1.543e2	388	1.173e2

Table 13
Execution time in seconds and number of iterations for computing $L^d\mathbf{b}$, where $\mathbf{b}$ is a vector with all ones (undirected case).

Network	ILU		Block - ILU-8		Block - ILU-16		Block - ILU-32		AMG	
	Its	Time	Its	Time	Its	Time	Its	Time	Its	Time
roadNet-CA	2644	4.372e2	2795	4.492e2	2949	4.677e2	3154	4.980e2	1482	9.073e2
roadNet-PA	1793	1.697e2	1931	1.738e2	1979	1.772e2	2110	1.885e2	892	2.770e2
roadNet-TX	3272	3.638e2	3457	3.836e2	3600	3.936e2	3711	4.051e2	1152	4.592e2
as-Skitter	163	5.277e1	265	6.181e1	306	6.623e1	316	6.535e1	+	+
hollywood-2009	18	1.008e2	42	6.675e1	46	5.364e1	47	4.409e1	302	1.262e3
packing-500x100x100-b050	340	1.039e2	382	1.159e2	395	1.168e2	411	1.223e2	+	+

Table 14
Execution time in seconds for computing L^d and $L^d\mathbf{b}$, where $\mathbf{b}$ is a vector with all ones, through the direct algorithm.

Network	n	Edges	L^d	$L^d\mathbf{b}$
email	1133	10902	2.493e-1	3.997e-2
data	2851	30186	1.902e0	1.012e-1
cage9	3534	41594	1.636e1	6.460e-1
uk	4824	13674	5.072e0	2.113e-1
power	4941	13188	4.905e0	2.101e-1
cell1	7055	30082	1.153e1	3.205e-1
wing_nodal	10937	150976	2.217e2	3.918e0

Table 15
Estimation of the ith diagonal entry of the absorption inverse matrix.

Network	n	Edges	i	Its	Time	Rel. error ub
email	1133	10902	10	11	2.054e-1	6.974e-3
data	2851	30186	100	67	1.548e1	9.863e-3
power	4941	13188	20	165	2.240e2	9.755e-3
wing-nodal	10937	150976	5	28	2.448e0	9.353e-3
cs4	22499	87716	50	55	1.037e1	9.811e-3
as-22July06	22963	96872	500	72	2.245e1	9.550e-3
wing	62032	243088	1	74	2.678e1	9.954e-3
fe_tooth	78136	905182	3	72	2.945e1	9.406e-3
fe_rotor	99617	1324862	2	146	1.775e2	9.807e-3
fe_ocean	143437	819186	30	215	4.484e2	9.826e-3

Example II. Computation of L^d and $L^d\mathbf{b}$ through the direct algorithm. Let us consider the undirected networks *email*, *data*, *uk*, *power*, *wing-nodal* and the directed networks *cage9* and *cell1*. In Table 14 we report the execution time in seconds for computing L^d and $L^d\mathbf{b}$, through the direct algorithm. The first three columns of the Table display the characteristics of the networks, i.e. the name, the number of nodes n and the number of edges of each network. We remark that L^d is generally a dense matrix.

From Table 14 we can see that for networks of moderate size the direct methods are sufficient for the computation of the absorption inverse L^d and of the quantity $L^d\mathbf{b}$. In particular, we observe that $L^d\mathbf{b}$ is computed in less than half second of CPU time, for graphs with up to 5000 nodes, and less than 5 seconds of CPU time, for graphs with up to 10000 nodes.

Example III. Estimation of individual entries of the absorption inverse matrix. In this Example, we estimate individual entries of the absorption inverse matrix for undirected graphs through the method described in section 6 by using Gauss-Radau quadrature rule. In Table 15 we report the iterations of the algorithm and the execution time in seconds for computing certain diagonal elements of the tested networks as well as an upper bound for the relative error that is attained for this estimation, given by the absolute value of the difference of the upper and lower bound for the diagonal element divided by the absolute value of its lower bound. In this Table we keep the upper bound for the relative error less that 10^{-2}.

In Table 15 we can see that individual entries of the test networks can be approximated using the Gauss quadrature rules and the Lanczos algorithm. In particular, using the Gauss-Radau rule we obtain upper and lower bounds for these entries which also give an upper bound for the relative error of this approximation, in a satisfactory execution time. It should be observed that the estimation of the ith diagonal entry is independent of the estimation of the jth diagonal entry; therefore, multiple diagonal entries can be computed in parallel.

8. Conclusions

We have described and compared different algorithms for the computation of the absorption inverse and related quantities. Direct and iterative methods with various preconditioners have been tested and compared for networks of various types. Techniques for estimating individual entries of the absorption inverse have been also discussed.

Based on the numerical methods developed, we also studied different centrality measures for graphs with absorption. These measures are compared with the one proposed in [13] for various graphs such as the star graph, the cycle and the path graph. The proposed centrality measures take into consideration both the absorption rates and the structure of the underlying graph and can be applied also when the absorption rates are all equal.

Furthermore, we have considered the case where the graph undergoes the addition (or deletion) of an edge. In this case the absorption inverse can be efficiently computed by updating the absorption inverse of the initial graph with a rank-two change.

Based on the numerical experiments performed, we can conclude that the proposed methods are efficient and provide easily applied tools for handling large graphs with several million nodes and edges.

Conflict of interest statement

There is no conflict of interest.

Acknowledgements

The authors acknowledge Junyuan Lin for sharing the AMG code. The work of M.B. was supported in part by the US National Science Foundation (grant DMS-1719518). P.F. would like to acknowledge financial support from the Foundation for Education and European Culture (IPEP) as well as from the Department of Mathematics and Computer Science of Emory University.

References

[1] Z. Bai, M. Fahey, G.H. Golub, Some large-scale matrix computation problems, J. Comput. Appl. Math. 74 (1996) 71–89.

[2] M. Benzi, A direct projection method for Markov chains, Linear Algebra Appl. 386 (2004) 27–49.
[3] M. Benzi, Preconditioning techniques for large linear systems: a survey, J. Comput. Phys. 182 (2002) 418–477.
[4] A. Berman, R.J. Plemmons, Nonnegative Matrices in the Mathematical Sciences, Society for Industrial and Applied Mathematics, Philadelphia, 1994.
[5] J.J. Buoni, Incomplete factorization of singular M-matrices, SIAM J. Algebr. Discrete Methods 7 (2) (1986) 193–198.
[6] S.L. Campbell, C.D. Meyer, Generalized Inverses of Linear Transformations, Pitman Publishing Ltd., London, 1979, Reprinted by Dover Publishing Co., New York, 1991.
[7] E. Estrada, The Structure of Complex Networks, Oxford University Press, Oxford, UK, 2011.
[8] E. Estrada, J.A. Rodríguez-Velázquez, Subgraph centrality in complex networks, Phys. Rev. E 71 (2005) 056103.
[9] R.E. Funderlic, R.J. Plemmons, Updating LU factorizations for computing stationary distributions, SIAM J. Algebr. Discrete Methods 7 (1) (1986) 30–42.
[10] R.E. Funderlic, R.J. Plemmons, LU decomposition of M-matrices by elimination without pivoting, Linear Algebra Appl. 41 (1981) 99–110.
[11] G.H. Golub, G. Meurant, Matrices, Moments and Quadrature with Applications, Princeton University Press, Princeton, 2010.
[12] X. Hu, J. Lin, L. Zikatanov, An adaptive multigrid method based on path cover, arXiv:1806.07028, 19 June, 2018.
[13] K.A. Jacobsen, J.H. Tien, A generalized inverse for graphs with absorption, Linear Algebra Appl. 537 (2018) 118–147.
[14] E.F. Kaasschieter, Preconditioned conjugate gradients for solving singular systems, J. Comput. Appl. Math. 24 (1988) 265–275.
[15] L. Katz, A new status index derived from sociometric data analysis, Psychometrika 18 (1953) 39–43.
[16] A.N. Langville, C.D. Meyer, Google's PageRank and Beyond: The Science of Search Engine Rankings, Princeton University Press, Princeton, NJ, 2006.
[17] O.E. Livne, A. Brandt, Lean Algebraic Multigrid (LAMG): fast graph Laplacian linear solver, SIAM J. Sci. Comput. 34 (2012) B499–B522.
[18] K. Morikuni, M. Rozloznik, On GMRES for singular EP and GP systems, SIAM J. Matrix Anal. Appl. 39 (2018) 1033–1048.
[19] M.E.J. Newman, Networks: An Introduction, Cambridge University Press, Cambridge, UK, 2010.
[20] Y. Saad, M. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Statist. Comput. 7 (3) (1986) 856–869.
[21] The SuiteSparse Matrix Collection, https://sparse.tamu.edu/.
[22] J.H. Tien, Z. Shuai, M.C. Eisenberg, P. van den Driessche, Disease invasion on community networks with environmental pathogen movement, J. Math. Biol. 70 (5) (2015) 1065–1092.
[23] J.C. Urschel, X. Hu, J. Xu, L. Zikatanov, A cascadic multigrid algorithm for computing the Fiedler vector of graph Laplacians, J. Comput. Math. 33 (2) (2015) 209–226.
[24] P. Van Mieghem, Graph Spectra for Complex Networks, Cambridge University Press, 2011.