Numerical Algorithms
https://doi.org/10.1007/s11075-025-02012-0

RESEARCH

A message-passing approach to obtain the trace of matrix functions with applications to network analysis

Grover Enrique Castro Guzman[1] · Peter Florian Stadler[2,3,4,5,6] · Andre Fujita[1,7]

Received: 8 May 2024 / Accepted: 14 January 2025

Abstract
Graphs have become a commonly used model to study technological, biological, and social systems. Various methods have been proposed to measure graphs' structural and dynamical properties, providing insights into the fundamental processes and interactions that govern the behavior of these systems. Matrix functions are powerful mathematical tools for assessing vertex centrality, communicability, and diffusion processes. Let $\mathbf{M}$ be the adjacency matrix of a weighted undirected graph. Then, the trace of matrix functions, $\mathbf{tr}(\boldsymbol{f}(\mathbf{M}))$, provides insights into global network structural and dynamical properties. Although $\mathbf{tr}(\boldsymbol{f}(\mathbf{M}))$ can be computed using the diagonalization method for graphs with a few thousand vertices, this approach is impractical for large-scale networks due to its computational complexity. Here, we present a message-passing method to approximate $\mathbf{tr}(\boldsymbol{f}(\mathbf{M}))$ for graphs with short cycles that runs in linear time up to logarithmic terms. We compare our proposal with the state-of-the-art approach through simulations and real-world network applications, achieving comparable accuracy in less time.

Keywords Message-passing · Matrix functions · Cauchy integrals · Lanczos · Chebyshev

1 Introduction

Graphs are a commonly used model for systems comprising interacting constituent parts. The application domains range from brain function, gene regulation, and ecosystems in biology to social interactions and technological systems. Many methods have been proposed to measure graphs and networks' global structural and dynamic properties [1–3]. In undirected graphs, the adjacency matrix $\mathbf{A}$, the Laplacian matrix $\mathbf{L}$, and the normalized Laplacian matrix $\mathcal{L}$ are symmetric. Let $\mathbf{M}$ be a matricial graph representation. Then, we can express some graph quantities using $\mathrm{tr}(f(\mathbf{M}))$. For example, the adjacency matrix allows us to compute the number of closed walks of length $k > 0$ ($f(z) = z^k$), the number of closed walks of length at most k ($f(z) = \sum_{j=0}^{k} z^j$), the Estrada index [4] or "index of the degree of folding" ($f(z) = \exp(z)$), and the number

Extended author information available on the last page of the article

of closed walks of odd and even length ($f(z) = \sinh(z)$ and $f(z) = \cosh(z)$, respectively) [5]. A complete list of matrix functions for network analysis can be found in [2].

The Laplacian matrix $\mathbf{L}$ provides key insights into the structure of a connected graph. This matrix is symmetric and positive semi-definite, and the multiplicity its zero eigenvalue corresponds to the number of connected components in the graph [6]. One of the most prominent measures derived from this matrix is the number of spanning trees. According to Kirchhoff's theorem, the number of spanning trees $t(G)$ can be computed from the non-zero eigenvalues of $\mathbf{L}$ as $t(G) = \lambda_2\lambda_3 \ldots \lambda_n/n$, where n is the number of vertices, and $\lambda_2, \lambda_3, \ldots, \lambda_n$ are the non-zero eigenvalues. Applying the natural logarithm on both sides yields $\ln(t(G)) = -\ln(n) + \sum_{i=2}^{n} \ln(\lambda_i)$ thus $f(z) = \ln z$. Another important measure derived from $\mathbf{L}$ is the Kirchhoff index, given by $Kf(G) = n \sum_{i=2}^{n} \frac{1}{\lambda_i}$, which leads to $f(z) = 1/z$. The Kirchhoff index is related to molecular properties, such as structural radius, viscosity, and the radius of gyration [7–9].

The practical computation of these measures naturally leads to classical problems of numerical linear algebra: solving sparse linear systems, eigenvalues and eigenvectors computations, and matrix functions evaluation [1, 2]. In particular, the computation of the trace of matrix functions is of interest because we can obtain graph measures such as communicability, centrality, bipartivity, and entropy, among others [2, 10, 11]. Here, we focus on the computation of the trace of a matrix function corresponding to an undirected graph. A straightforward idea is to obtain the exact value of the trace using diagonalization (LU or Cholesky decomposition). The problem is that these algorithms have a computational complexity of $\mathcal{O}(n^3)$ and require $\mathcal{O}(n^2)$ memory, which is prohibitive for the large graphs we are interested in.

Thus, more efficient methods have been developed to avoid diagonalization of the matrix. [12] proposed a stochastic Lanczos quadrature (SLQ) method for approximating the trace of matrix functions, demonstrating that it performs better than methods using Taylor series [13] and Chebyshev polynomials [14, 15]. This method has a computational complexity of $\mathcal{O}(qn_v|E|)$ (q is the number of quadratures, and n_v is the number of random vectors used), much lower than the diagonalization approach. Recently, [11] proposed a block approach of [12] with the same time complexity. The advantage of the method proposed by [11] is that we can run it on modern computer architectures (i.e., multi-cores). However, it only works with completely monotonic functions. In contrast, [12] only requires f to be analytic inside a closed interval. The main disadvantage of these methods is that they require access to the entire matrix to perform matrix-vector operations (global computations).

Message-passing (MP) algorithms are computational methods used to solve problems on graphical models, such as Bayesian networks and Markov Random Fields [16, 17]. These algorithms rely on local updates and iterative computations to estimate marginal probabilities and solve optimization problems efficiently. The main advantage is their scalability to large-scale graphs, as they operate using local information, thus avoiding global computations. This makes them particularly suitable for applications in error-correcting codes, probabilistic inference, and optimization problems, is particular for graphs with sparse structures [18]. Additionally, message-passing algorithms often converge quickly, even in high-dimensional systems [18, 19].

However, an essential requirement for such methods is that the graph has a locally tree-like structure; their performance usually degrades when graphs have short loops [20].

Here, we propose a message-passing method that leverages local computations to compute the trace of matrix functions, providing a scalable alternative for large-scale network analysis. Our approach is not limited to completely monotonic functions and utilizes only the local neighborhood of each vertex [16, 17]. Moreover, we use the definition of local neighborhoods introduced in [20] to develop a method for graphs with short loops. The methodology begins with describing the basic concepts (Section 2). Next, we derive message-passing equations for networks with short loops (Section 3). To validate our proposal, we compare it against the SLQ method in computer simulations and two real-world networks: co-authorship and PGP. These comparisons assess accuracy and computational efficiency, demonstrating the efficacy of our approach (Section 4).

2 Graphs, matrix functions, and Cauchy integrals

Let $G = (V, E)$ be a graph with a vertex set $V = \{1, 2, \ldots, n\}$, where $n = |V|$ is the number of vertices, and an edge set $E \subseteq V \times V$ with $m = |E|$ edges, along with a weight function $w : E \to \mathbb{C}$. Subsequently, we use $V(.)$ and $E(.)$ to refer to a graph's set of vertices and edges, respectively. Throughout, we will assume that G is undirected and that may contain self-loops, i.e., $(u, v) \in E(G)$ iff $(v, u) \in E(G)$ and $w(u, v) = \bar{w}(v, u)$ for all $(u, v) \in E(G)$, where $\bar{w}(v, u)$ is the complex conjugate of $w(v, u)$. We are interested, in particular, in sparse graphs. While there is no generally accepted definition of sparse graphs, we assume that the average degree $m/2n$ is constant or grows only slowly with n, e.g. as some power of $\ln n$. Let $\mathbf{M}$ be the $n \times n$ adjacency matrix of G defined as follows

$$\mathbf{M}_{ij} = \begin{cases} 0, \text{ if } (i, j) \notin E(G), \\ w(i, j), \text{ otherwise.} \end{cases}$$

Since G is undirected, the matrix $\mathbf{M}$ is Hermitian, and thus it has the eigendecomposition $\mathbf{M} = \mathbf{Q}\Lambda\mathbf{Q}^H$, where the columns of $\mathbf{Q}$ contain eigenvectors of $\mathbf{M}$, $\mathbf{Q}^H$ is the conjugate transpose of $\mathbf{Q}$, and $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ is a diagonal matrix that contains the eigenvalues (spectrum) $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ of $\mathbf{M}$.

Let f be a scalar function that is either polynomial, rational, or defined by a convergent power series [11, 21], within a closed interval containing the spectrum of $\mathbf{M}$ [21]. Then, a matrix function is defined as $f(\mathbf{M}) = \mathbf{Q}f(\Lambda)\mathbf{Q}^H$, where $f(\Lambda) = \text{diag}(f(\lambda_1), \ldots, f(\lambda_n))$, and the trace of $f(\mathbf{M})$ is given by:

$$\text{tr}(f(\mathbf{M})) = \sum_{i=1}^{n} f(\lambda_i).$$

The most straightforward approach to compute the trace of any matrix function $f(\mathbf{M})$ is to obtain the spectrum of $\mathbf{M}$ using LU or Cholesky decomposition.

An alternative way to compute $\mathrm{tr}(f(\mathbf{M}))$ is to use the Cauchy integral formula [14]:

$$f(\mathbf{M}) = \frac{1}{2\pi i} \oint_{\Gamma} f(z)(z\mathbf{I} - \mathbf{M})^{-1} dz, \tag{1}$$

where f is analytic inside the closed contour Γ that contains all eigenvalues of $\mathbf{M}$. The desired trace can then be written in the form [21]

$$\mathrm{tr}(f(\mathbf{M})) = \frac{1}{2\pi i} \oint_{\Gamma} f(z)\,\mathrm{tr}((z\mathbf{I} - \mathbf{M})^{-1}) dz. \tag{2}$$

Equation (2) requires a continuous integral that we can approximate using quadratures [21]. Among many quadrature rules (e.g., trapezoidal, circular, and ellipsoidal) [22], we chose the circular rule because it only requires the interval $[a, b]$ that encloses the eigenvalues (i.e., $a = \lambda_1$ and $b = \lambda_n$) [21]. Nonetheless, we must carefully choose the interval $[a, b]$ because f does not need to be defined on the entire complex plane. For example, $f(z) = \ln(z)$, which appears in the computation of the logarithm of the number of spanning trees, is not well-defined if the interval $[a, b]$ includes non-positive numbers. For the quantities mentioned in the introduction, the interval $[a, b] = [\lambda_1, \lambda_n]$ can be used for the adjacency matrix, $\mathbf{M} = \mathbf{A}$, while for both measures depending on the Laplacian matrix, $\mathbf{M} = \mathbf{L}$, an interval of the form $[a, b] = [\epsilon, \lambda_n]$ with a small $\epsilon > 0$ must be used to exclude the eigenvalue 0.

Given a suitable interval $[a, b]$ for the circular quadrature with q quadrature points z_j, we can calculate $\mathrm{tr}(f(\mathbf{M}))$ as follows:

$$\mathrm{tr}(f(\mathbf{M})) = \sum_{j=1}^{q} w_j f(z_j)\,\mathrm{tr}((z_j\mathbf{I} - \mathbf{M})^{-1}), \tag{3}$$

where z_j are quadrature points, and w_j are the quadrature weights. The major bottleneck in obtaining $\mathrm{tr}(f(\mathbf{M}))$ is the computation of $\mathrm{tr}((z_j\mathbf{I} - \mathbf{M})^{-1})$. Since obtaining an exact solution is challenging and computationally expensive, approximation methods such as the Taylor series [13], Chebyshev polynomials [14, 15], or SLQ [12] have been proposed. Still, all these methods require matrix-vector multiplications and might also need the largest and smallest eigenvalues [12]. The SLQ method is more accurate and faster than the other methods and requires $\mathbf{M}$ to be Hermitian.

In the next section, we present a message-passing (MP)-based method with the same requirement as SLQ: $\mathbf{M}$ to be Hermitian.

3 Message-passing equations

The authors in [20] introduced a set of message-passing equations to compute the probability density of the eigenvalues of a weighted undirected graph's adjacency and Laplacian matrices. Since $\mathbf{M}$ is the weighted adjacency matrix of G, that approach

can also be used to compute the eigenvalues of $\mathbf{M}$ and, consequently, to compute $\mathrm{tr}((z_j\mathbf{I}-\mathbf{M})^{-1})$.

Let $\rho(\mathbf{M})$ be the spectral radius of $\mathbf{M}$, and let the weight of a walk in G be the product of the weights of the edges it traverses. The message-passing equations obtained in [20] serve to compute the diagonal elements (i.e., the sum of the weights of the closed walks from any vertex) of $(z_j\mathbf{I}-\mathbf{M})^{-1}$ by considering it as a geometric series, that is, $(z_j\mathbf{I}-\mathbf{M})^{-1} = (1/z_j)\sum_{l=0}^{\infty}(\mathbf{M}/z_j)^l$. This series clearly converges for $|z_j| > \rho(\mathbf{M})$.

Let $\mathcal{N}_i^r$ be the sub-graph induced by the edges and vertices included in any cycle from vertex i of length at most $r+2$. Also, let $\mathcal{N}_{j\setminus i}^r$ be the sub-graph induced by the edges and vertices included in any cycle from vertex j, of length at most $r+2$, without considering the edges in $\mathcal{N}_i^r$. Figure 1 shows an example of $\mathcal{N}_i^r$ for different values of r.

A walk w of length $l \geq 0$ in G is a sequence $w = (w_0, w_1, \ldots, w_l)$ of vertices $w_k \in V(G)$ such that $(w_k, w_{k+1}) \in E(G)$. We define the weight of a walk (walk-sum) as the product of edge weights along the walk:

$$\phi(w) = \prod_{k=1}^{l} \mathbf{M}_{w_{k-1}w_k}.$$

The idea proposed in [20] is to use the local neighborhoods of a vertex to compute $(z_j\mathbf{I}-\mathbf{M})^{-1}_{ii}$ for all $i \in V(G)$ (the sum of the walk-sums of walks of any length that start and end in the same vertex).

Let $n_{i\leftarrow j} = |V(\mathcal{N}_{j\setminus i}^r)| - 1$, $n_i = |V(\mathcal{N}_i^r)| - 1$, and $\mathbf{v}_{i\leftarrow j} \in \mathbb{C}^{n_{i\leftarrow j}\times 1}$ be a column vector such that $\mathbf{v}_{j\leftarrow j,k} = \mathbf{M}_{j,k}$ if $(j,k) \in E(\mathcal{N}_{j\setminus i}^r)$ and 0 otherwise. Denote by $\mathbf{A}^{i\leftarrow j} \in \mathbb{C}^{n_{i\leftarrow j}\times n_{i\leftarrow j}}$ the adjacency matrix of $\mathcal{N}_{j\setminus i}^r$ after removing vertex j. Then, we have

$$\mathbf{A}^{i\leftarrow j}_{k,l} = \begin{cases} \mathbf{M}_{k,l}, & \text{if } k,l \neq j \text{ and } (k,l) \in E(\mathcal{N}_{j\setminus i}^r), \\ 0, & \text{Otherwise.} \end{cases}$$

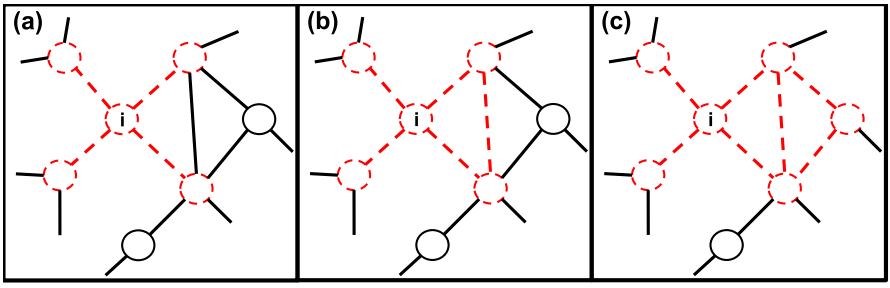

Fig. 1 Local neighborhood of vertex i ($\mathcal{N}_i^r$) is shown with dashed lines. In contrast, the neighborhood of vertex j ($\mathcal{N}_{j\setminus i}^r$), excluding edges in $\mathcal{N}_i^r$, is depicted with dotted lines for different values of r. (a) $r = 0$. (b) $r = 1$. (c) $r = 2$

Let $H_{i\leftarrow j}(z)$ be the sum of the walk-sums of all walks that visit j only at the start and end, without using the edges of $\mathcal{N}_i^r$ [20]. Let $\mathbf{D}^{i\leftarrow j}(z) \in \mathbb{C}^{n_{i\leftarrow j}\times n_{i\leftarrow j}}$ be a diagonal matrix, such that:

$$\mathbf{D}^{i\leftarrow j}_{k,k}(z) = \begin{cases} z - H_{j\leftarrow k}(z), & \text{if } k \in V(\mathcal{N}^r_{j\setminus i}) \setminus \{j\}, \\ 0, & \text{Otherwise.} \end{cases}$$

Note that $(\mathbf{D}^{i\leftarrow j}(z) - \mathbf{A}^{i\leftarrow j})^{-1}$ entries are the sum of all walk-sums whose start and end are in $V(\mathcal{N}^r_{j\setminus i}) \setminus \{j\}$. Thus, $H_{i\leftarrow j}(z)$ is given by [20]:

$$H_{i\leftarrow j}(z) = \mathbf{v}^H_{i\leftarrow j}(\mathbf{D}^{i\leftarrow j}(z) - \mathbf{A}^{i\leftarrow j})^{-1}\mathbf{v}_{i\leftarrow j} + \mathbf{M}_{j,j}. \tag{4}$$

This is called a message-passing equation because the value of $H_{i\leftarrow j}(z)$ is determined by the values (messages) of $H_{j\leftarrow k}(z)$, $\forall k \in V(\mathcal{N}^r_{j\setminus i})\setminus\{j\}$ (present in $\mathbf{D}^{i\leftarrow j}(z)$).

Note that (4) computes the sum of walk-sums of all closed walks of a vertex j when the edges of $\mathcal{N}_i^r$ are not considered. Now, we will obtain the sum of walk-sums of all closed walks of a vertex $i \in V(G)$. Let $\mathbf{v}_i \in \mathbb{C}^{n_i\times 1}$ be a column vector such that $\mathbf{v}_{i,k} = \mathbf{M}_{i,k}$ if $(i,k) \in E(\mathcal{N}_i^r)$ and 0 otherwise, and $\mathbf{A}^i \in \mathbb{C}^{n_i\times n_i}$ be the adjacency matrix of $\mathcal{N}_i^r$ after removing vertex i, i.e.,

$$\mathbf{A}^i_{k,l} = \begin{cases} \mathbf{M}_{k,l}, & \text{if } k,l \neq i \text{ and } (k,l) \in E(\mathcal{N}_i^r), \\ 0, & \text{Otherwise.} \end{cases}$$

Let $\mathbf{D}^i(z) \in \mathbb{C}^{n_i\times n_i}$ be a diagonal matrix defined as follows:

$$\mathbf{D}^i_{k,k}(z) = \begin{cases} z - H_{i\leftarrow k}(z), & \text{if } k \in V(\mathcal{N}_i^r) \setminus \{i\}, \\ 0, & \text{Otherwise.} \end{cases}$$

Note that $(\mathbf{D}^i(z) - \mathbf{A}^i)^{-1}$ entries are the sum of all walk-sums whose start and end are in $V(\mathcal{N}_i^r) \setminus \{i\}$. Following [20], we define $H_i(z)$ (the sum of the walk-sums of all walks that visit vertex i only at the start and end [20]) as follows:

$$H_i(z) = \mathbf{v}_i^H(\mathbf{D}^i(z) - \mathbf{A}^i)^{-1}\mathbf{v}_i + \mathbf{M}_{i,i}. \tag{5}$$

Since we require the sum of the walk-sums of all closed walks at i, not just those that visit vertex i at the start and end, we compute it as $(z - H_i(z))^{-1}$. Then, we have that:

$$\operatorname{tr}((z\mathbf{I} - \mathbf{M})^{-1}) = \sum_{i=1}^{n} \frac{1}{z - H_i(z)}. \tag{6}$$

Finally, we obtain $\mathrm{tr}(f(\mathbf{M}))$ as follows:

$$\begin{aligned}\mathrm{tr}(f(\mathbf{M})) &= \sum_{j=1}^{q} w_j f(z_j)\,\mathrm{tr}((z_j\mathbf{I}-\mathbf{M})^{-1}),\\ &= \sum_{j=1}^{q} w_j f(z_j) \sum_{i=1}^{n} \frac{1}{z_j - H_i(z_j)}.\end{aligned} \tag{7}$$

To show that our method converges, we extend the convergence conditions for message-passing in Gaussian Markov Networks [23], and present them in Propositions 1 and 2. We say that the matrix $\mathbf{M}$ is *walk-summable* (WS) [23] if the sum over all walks w from i to j ($\forall i, j \in V(G)$):

$$\sum_{w:i\to j} \phi(w),$$

converges to the same value for all possible summation orders. Thus, the message-passing (4) and (6) converge whenever $\mathbf{M}/z_j, \forall 1 \le j \le q$ are *walk-summable* [23].

Proposition 1 (Walk-Summability – WS) *Let $z \in \{z_1, z_2, \ldots, z_q\}$ be a quadrature point, and $\mathbf{R} = \mathbf{M}/z$. Each of the following three conditions is equivalent to the walk-summability of $\mathbf{R}$:*

1. $\sum_{w:i\to j} |\phi(w)|$ converges for all $i, j \in V(G)$.
2. $\sum_{l=0}^{\infty} \tilde{\mathbf{R}}^l$ converges.
3. $\rho(\tilde{\mathbf{R}}) < 1$.

Proposition 2 (WS Necessary Conditions) *Let $z \in \{z_1, z_2, \ldots, z_q\}$ be a quadrature point , and $\mathbf{R} = \mathbf{M}/z$. The walk-summability of $\mathbf{R}$ implies the following two equivalent conditions:*

1. $\rho(\mathbf{R}) < 1$,
2. $\sum_{k=0}^{\infty} \mathbf{R}^k = (\mathbf{I}-\mathbf{R})^{-1}$.

The proofs are in Appendix A. Notice that $H_{i\leftarrow j}(z)$ and $H_i(z)$ are entirely independent of f. Hence, when computing $\mathrm{tr}(f_k(\mathbf{M}))$ for a constant matrix $\mathbf{M}$ and different functions f_k, $k \in \{1, \ldots, p\}$ with $p > 0$, it is sufficient to compute $H_{i\leftarrow j}(z)$ and $H_i(z)$ once. These values can then be reused for each f_k.

Now, we will analyze the time and space complexity of our proposed method. Let

$$n_{\max} = \max\left\{ n_{i\leftarrow j} \middle| i \in V(G) \text{ and } j \in V(\mathcal{N}_i^r) \setminus \{i\} \right\}, \tag{8}$$

Let $\Delta = \max_i |\{j \in V(G) | (i, j) \in E(G)\}|$ be the largest degree of G. It is not difficult to see that $\Delta \le n_{\max} \le \Delta^{r+1}$. Moreover, we denote by d be the diameter of G. Given initial values for $H_{i\leftarrow j}(z)$ (e.g., zero), we run (4) until convergence (i.e., until the absolute difference between its previous and new value is smaller than a

given tolerance, such as 10^{-16}), as detailed in Algorithm 1. According to heuristics [19], the convergence of (4) is attained in d iterations. Then, the time complexity to compute $H_{i\leftarrow j}(z)$ is $\mathcal{O}(n_{\max} d|E|)$ when $r = 0$ and $\mathcal{O}(n_{\max}^3 d|E|)$ otherwise. After achieving convergence using (4), we use (5) to compute $H_i(z), \forall i \in V(G)$. This computation has a time complexity of $\mathcal{O}(n_{\max} n)$ when $r = 0$ and $\mathcal{O}(n_{\max}^3 n)$ otherwise. Therefore, to obtain $\mathrm{tr}(f(\mathbf{M}))$, we employ Algorithm 2, which has a time complexity of $\mathcal{O}(n_{\max} d|E|)$ for $r = 0$ and $\mathcal{O}(n_{\max}^3 d|E|)$ otherwise. The time complexity for $r = 0$ is different because the matrices in (4) and (5) are diagonal. The space complexity of our method is $\mathcal{O}(n_{\max}^2|E|)$ regardless of the value of r. As long as the vertex degree is bounded logarithmically we therefore obtain a running time bound of the form $O(dn \ln^{k(r)} n)$ where $k(r)$ grows linearly with r.

Algorithm 1 Compute $H_i(z)$ for all $i \in V$

Require: $\mathcal{N}_i^r \forall i \in V, \mathcal{N}_{i\leftarrow j}^r \forall i, j$ with $i \in V$ and $j \in \mathcal{N}_i^r, z, tol = 10^{-16}$
  $l \leftarrow \sum_{i=1}^n |V(\mathcal{N}_i^r)|$
  $H_{i\leftarrow j} \leftarrow 0, \forall i, j$ with $i \in V$ and $j \in V(\mathcal{N}_i^r)$
  while *true* **do**
    $error \leftarrow 0$
    for $i \in V$ **do**
      for $j \in V(\mathcal{N}_i^r)$ **do**
        $prev\_val \leftarrow H_{i\leftarrow j}(z)$
        Update $H_{i\leftarrow j}(z)$ using (4).
        $error \leftarrow error + |prev\_val - H_{i\leftarrow j}(z)|$
      end for
    end for
    $error \leftarrow error/l$
    if $error < tol$ **then**
      break
    end if
  end while
  for $i \in V$ **do**
    Compute $H_i(z)$ using (5).
  end for
  return $H_i(z) \forall i \in V$

3.1 Local measures

We can straightforwardly extend the message-passing approach to local measures. The contribution $\gamma_i^f = f(\mathbf{M})_{i,i}$ of a single vertex i to a global measure $\mathrm{tr}(f(\mathbf{M}))$ is sometimes called the "centrality" of i [2]. We can obtain it from (7) by rearranging the sums:

$$\mathrm{tr}(f(\mathbf{M})) = \sum_{i=1}^{n} \sum_{j=1}^{q} \frac{w_j f(z_j)}{z_j - H_i(z_j)} =: \sum_{i=1}^{n} \gamma_i^f. \tag{9}$$

Using (9), one can estimate, for example, the number of closed walks of length k that pass through i using $f(z) = z^k$. Meanwhile, we can obtain all even and odd closed

Algorithm 2 Compute $\mathrm{tr}(f(\mathbf{M}))$

Require: $\mathbf{M}$, f, q, r
  Obtain the graph $G = (V, E)$ corresponding to matrix $\mathbf{M}$.
  Obtain $\mathcal{N}_i^r, \mathcal{N}_{i\leftarrow j}^r$ of G.
  Compute the q quadratures $(z_1, w_1), (z_2, w_3), \ldots, (z_q, w_q)$.
  $trace \leftarrow 0$
  for $j \in 1, 2, \ldots, q$ **do**
    Use algorithm 1 to obtain $H_i(z_j) \forall i \in V$.
    for $i \in V$ **do**
      $trace \leftarrow trace + \frac{w_j f(z_j)}{z_j - H_i(z_j)}$
    end for
  end for
  return $\Re(trace)$

walks using $f(z) = \cosh(z)$ and $f(z) = \sinh(z)$, respectively. Local measures derived from the (un)normalized Laplacian can be obtained and interpreted similarly.

4 Results

We demonstrate the accuracy and efficacy of our method on both simulated and real-world graphs. As a baseline, we use the SLQ method [12]. We report average and 95% confidence intervals estimated from 10 replicates for all experiments. We generated the graphs ($n = 40\,000$ vertices) using three random graph models.

1. The **configuration model** [24] generates graphs with a given degree sequence. The resulting graphs are locally tree-like, and we expect our method to work well with $r = 0$. We choose half of the vertices with degree 5 and the other half with degree 10.
2. The **Watts-Strogatz model** [25] generates small-world graphs with an average vertex degree k. The algorithm starts from a regular lattice, and with probability p, each edge is randomly rewired to a different vertex. We chose $k = 8$ and $p = 0.1$. The resulting graphs present many short cycles. We expect that the approximation's accuracy increases with the parameter r.
3. The **Barabási-Albert model** [26, 27] generates networks using a preferential attachment mechanism. We set the average degree to $k = 2$ and a scaling parameter to $\gamma = 1.2$. The matrix representations of these graphs are ill-conditioned [28]. Consequently, we expect the SLQ method does not obtain good approximations [29].

In addition to the randomly generated graphs, we test our approach on two real-world networks.

The **co-authorship network** (https://snap.stanford.edu/data/ca-CondMat.html) comprises 23 134 vertices and 93 497 edges [30] representing co-authorship relations in condensed matter physics.
The **PGP network** (http://konect.cc/networks/arenas-pgp/) represents trust relations among users of the PGP encryption software comprising 10 681 vertices and 24 316 edges [31].

All computations were performed on an Intel Xeon Silver 4116 CPU 2.10GHz and 160GB of memory. The code was parallelized to run in 30 cores.

For benchmarking, we selected the number of closed walks of length three, i.e., $f(z) = z^3$, and the logarithm of the number of spanning trees, i.e., $f(z) = \ln z$, as examples. Counting triangles helps identify local clustering and community structures, which are key for understanding collaboration and information flow [24]. On the other hand, spanning trees provide insights into the global connectivity and resilience of the network, which are critical for analyzing robustness and ensuring efficient communication pathways in complex systems [32]. To compute the number of spanning trees, the graph must be connected (*Matrix-Tree Theorem*) [33]. Then, we used the largest connected component of both real-world networks to calculate it. This condition is not required to compute the number of triangles.

The interval for integration must contain all eigenvalues of the adjacency matrix and all non-zero (and thus strictly positive) eigenvalues of the Laplacian matrix. In terms of the weighted vertex degree $d_i = \sum_j \mathbf{A}_{ij}$, the spectral radius of $\mathbf{A}$ satisfies $\rho \leq \max_{i\sim j} \sqrt{d_i d_j} \leq \max_i d_i$ [34]. For the Laplacian matrix $\mathbf{L}$, a well-known bound is $\rho \leq \max_{i\sim j}(d_i + d_j)$ [35]. For a connected graph, the 2nd-smallest eigenvalue of $\mathbf{L}$, also known as the algebraic connectivity, is positive. Lower bounds for the algebraic connectivity are known for unweighted graphs; see, e.g., [36]. However, a convenient lower bound for weighted graphs is not available.

Since neither $f(z) = z^3$ nor $f(z) = \ln z$ is completely monotonic (they do not have derivatives $f^{(l)}(x)$ for all $l = 1, 2, 3, \ldots$ such that $(-1)^l f^{(l)}(x) \geq 0$ [37]), the estimation procedure described in [11] is not applicable. Thus, we compare only against the SLQ method, using the same parameters recommended in [12]. Let $\mu = \mathrm{tr}(f(\mathbf{M}))$ be the exact value obtained with the diagonalization method, and $\hat{\mu}$ be the numerical estimate. Then, we measure the accuracy as a signed percent deviation, i.e., $100(\hat{\mu} - \mu)/\mu$. The sign of this error measure indicates whether the approximation method underestimates (negative) or overestimates (positive) the value of $\mathrm{tr}(f(\mathbf{M}))$.

The first experiment consists of computing the number of closed walks of length three and the logarithm of the number of spanning trees with different numbers of quadratures. We recommend using $r = 1$ and $q = 16$ quadratures, based on additional experiments presented in Appendix B. We observed that increasing the number of quadratures reduces the signed error, and $q = 16$ quadratures offer a favorable balance between minimizing signed percent error and maintaining low time complexity when computing the number of triangles and spanning trees. Moreover, for graphs generated by the configuration model, which are locally tree-like as the number of vertices increases [24], using $r = 0$ suffices. However, we achieved the best approximations for other graph models by accounting for primitive cycles of length 3, i.e., $r = 1$.

Table 1 summarizes the relative errors for the number of walks of length three and the logarithm of the number of spanning trees, respectively, using the message-passing approach with $r = 1$ and $q = 16$. We observe that the relative errors are comparable to the SLQ method. However, we obtain tighter confidence intervals for the message-passing approach in computing the number of triangles. It also yields tighter confidence intervals for the number of spanning trees for all graph models except the Barabási-Albert model. As expected, the SLQ method produces substantial

Table 1 Mean signed percent error and 95% confidence interval (in square brackets) for estimates of the number of triangles (closed walks of length three) and the logarithm of the number of spanning trees with the SLQ method and our message-passing approach with $r = 1$ and $q = 16$

	Triangles		Spanning Trees	
Model	SLQ	MP	SLQ	MP
Configuration	272.1 [-124.7;613.6]	195.1 [136;246]	4.8 [2.1;8.7]	0.0 [0.0;0.1]
Watts-Strogatz	-0.25 [-0.7;0.1]	0.0 [0.0;0.01]	0.0 [-7.9;4.6]	0.3 [0.3;0.3]
Barabási-Albert	-168.1 [-698.5;77.1]	192.2 [70.0;261.1]	-6.0 [-6.4;-5.6]	13.7 [10.9;15.9]

relative errors for graphs generated by the Barabási-Albert model. Unexpectedly, this is also the case for graphs generated by the configuration model. This occurs because the approximations obtained by the SLQ method are proportional to the weighted sum of the function $f(z)$ applied to the approximations of the eigenvalues. Thus, large eigenvalues dominate the final result. These errors will be more pronounced for large values of the scaling parameter of the Barabási-Albert model. In contrast, the errors in our message-passing method are related to the number of quadratures used.

Table 2 shows the results obtained using the SLQ and message-passing ($r = 1, q = 16$) methods. We can observe that the message-passing method is faster than the SLQ method when the configuration model generates the graph. Also, our proposal has a similar execution time for the other two graph models.

Figures 2 and 3 show the number of iterations required for $H_{i \leftarrow j}(z)$ for all i, j with $i \in V$ and $j \in \mathcal{N}_i^r$ to converge when estimating the number of triangles and the logarithm of the number of spanning trees. We set $r \in \{0, 1, 2\}$, $q = 16$, and z_8 (the eight quadrature point) for all the graph models (configuration, Watts-Strogatz, and Barabási-Albert). The y-axis of the plot shows the logarithm of the error term from Algorithm 1. We considered that $H_{i \leftarrow j}(z_8)$ converged when the error is smaller than $tol = 10^{-16}$. Our results show that Algorithm 1 required approximately 8-10 iterations to attain convergence, i.e., the expected diameter $\log(n)$ of the considered graph models [24]. For the configuration and Barabási-Albert models, the convergence is similar for any value of r, as expected, because these models generate graphs with

Table 2 Mean and standard deviation (between parentheses) of the time execution (in minutes) for estimates of the number of triangles (closed walks of length three) and the logarithm of the number of spanning trees with the SLQ method and our message-passing (MP) approach with $r = 1$ and $q = 16$

		Triangles		Spanning Trees	
Model	Exact	SLQ	MP	SLQ	MP
Configuration	171.3 (1.4)	21.3 (0.1)	10.5 (0.4)	21.5 (0.2)	9.8 (0.2)
Watts-Strogatz	171.6 (1.8)	36.5 (0.4)	44.6 (0.3)	36.4 (0.2)	38.3 (0.5)
Barabási-Albert	165.1 (5.3)	17.7 (0.2)	21.4 (4.5)	17.5 (0.2)	14.1 (2.8)

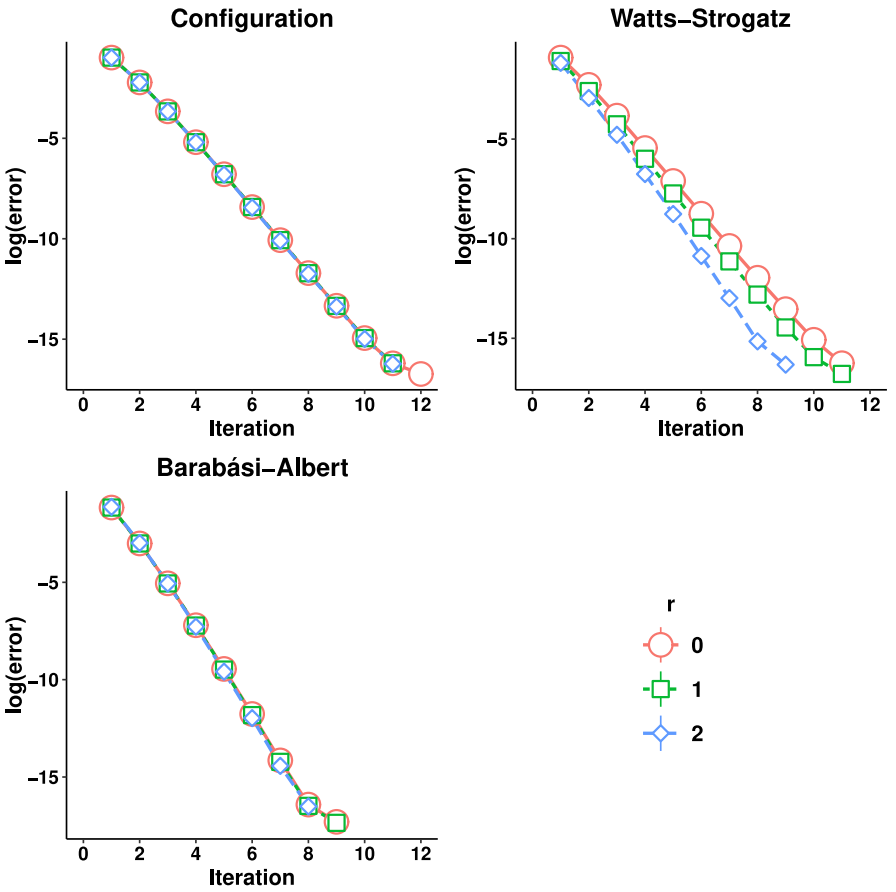

Fig. 2 The convergence behavior of $H_{i\leftarrow j}$ when estimating the number of triangles. We set $r \in \{0, 1, 2\}$, $q = 16$, and z_8 (the eighth quadrature point). We considered that $H_{i\leftarrow j}(z_8)$ converged when the error was less than 10^{-16}. Our algorithm converged in 8-10 iterations for the three random graph models (configuration, Watts-Strogatz, and Barabási-Albert). As expected, the convergence is faster for larger values of r

relatively few short cycles. In contrast, for the Watts-Strogatz model, the convergence depends on the values of r, as this model generates graphs with short cycles.

Table 3 shows the approximations of the number of triangles and spanning trees of the two real-world networks using the SLQ and message-passing ($r = 1, m = 16$) methods, respectively. We selected these parameters because they output the best approximation (Table 5). Our method provides better approximations than the SLQ method for computing the number of triangles. In contrast, our method overestimated the number of spanning trees. The reason for this is the number of quadratures we used to compute the number of spanning trees. Then, we increased the number of quadratures to obtain a better approximation. However, increasing the number of quadratures increases the running time (Table 6). The computational complexity can be estimated as $\mathcal{O}(n_{\max} d|E|q)$, where q is the number of quadrature points. Thus,

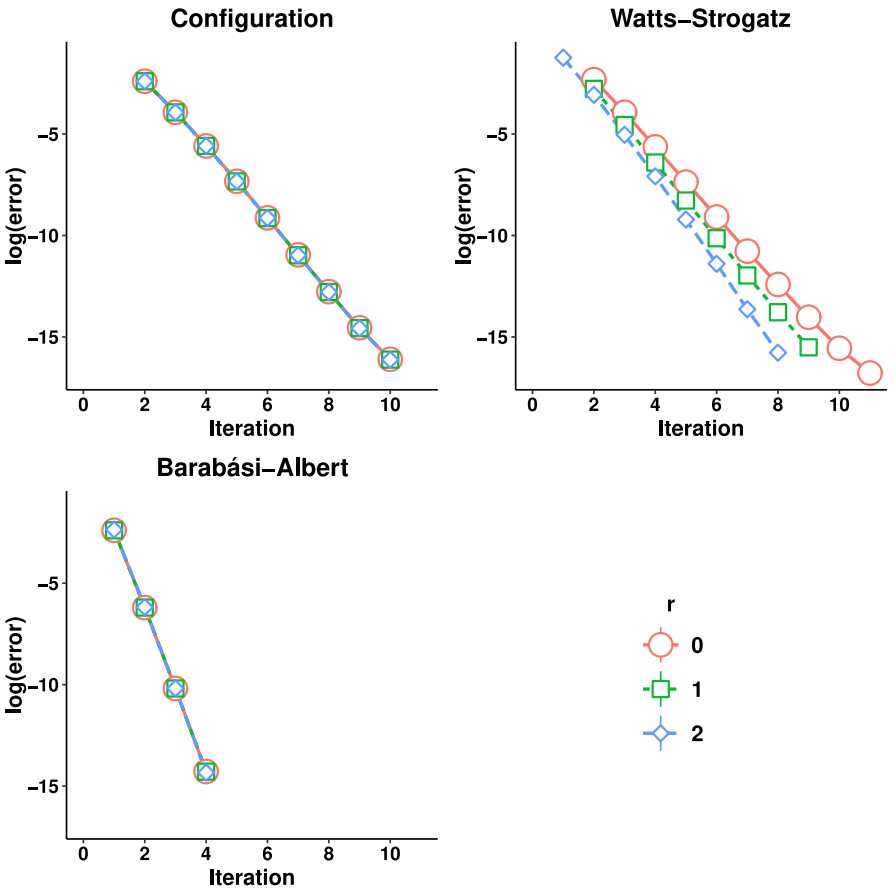

Fig. 3 The convergence behavior of $H_{i\leftarrow j}$ when estimating the number of spanning trees. We set $r \in \{0, 1, 2\}$, $q = 16$, and z_8 (the eighth quadrature point). We considered that $H_{i\leftarrow j}(z_8)$ converged when the error was less than 10^{-16}. Our algorithm converged in 8-10 iterations for the three random graph models (configuration, Watts-Strogatz, and Barabási-Albert). As expected, the convergence is faster for larger values of r

we expect to quadruplicate the running time by increasing q from 16 to 64. We also observed it empirically.

Table 4 displays the execution times (in minutes) for computing the number of triangles and spanning trees of two real-world networks using the SLQ and message-passing

Table 3 Number of triangles and spanning trees obtained with the exact, SLQ, and message-passing ($r = 1$, $q = 16$) methods of two real-world networks

Model	Triangles			Spanning trees		
	Exact	SLQ	MP	Exact	SLQ	MP
Co-authorship	1 048 160	1 032 010	1 048 680	27 102	32 858	37 424
PGP	328 728	344 778	329 540	6 899	8 147	9 828

Table 4 Number of triangles and spanning trees execution time (in minutes) obtained with the exact, SLQ, and message-passing ($r = 1, q = 16$) methods of two real-world networks

Model	Triangles			Spanning trees		
	Exact	SLQ	MP	Exact	SLQ	MP
Co-authorship	48.6	7.0	49.4	44.8	6.9	34.7
PGP	6.4	2.7	11.3	6.1	2.6	8.3

($r = 1, q = 16$) methods, respectively. The SLQ method is more efficient than ours despite similar execution times observed in simulations with random graph models.

The discrepancy arises from real-world networks containing vertices with numerous neighbors connected among them. Consequently, our message-passing method operates slower due to the computation of inverses of large matrices. For instance, the PGP network features a vertex with 259 interconnected neighbors, resulting in the message-passing method computing the inverse of a 259×259 matrix in each iteration. We could consider a parameter r for each vertex to mitigate this issue, as proposed in [38].

Finally, Figs. 4 and 5 present scatter plots comparing the exact and approximate values of the local measures for the number of triangles and spanning trees. The approximation for the local number of triangles is highly accurate for both real-world

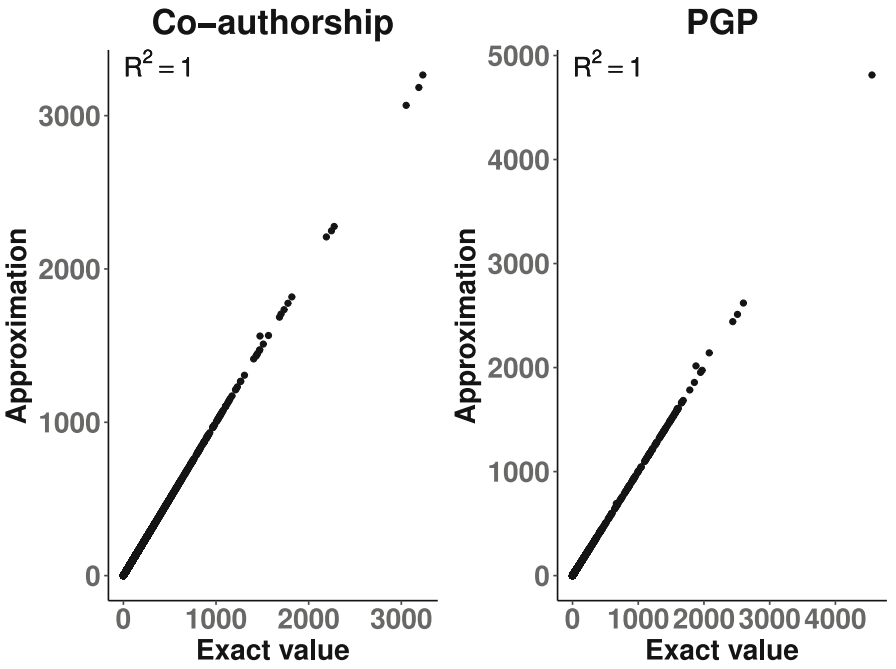

Fig. 4 Scatter plot of the Exact vs. Approximate value of the local measures for the number of triangles of two real-world networks

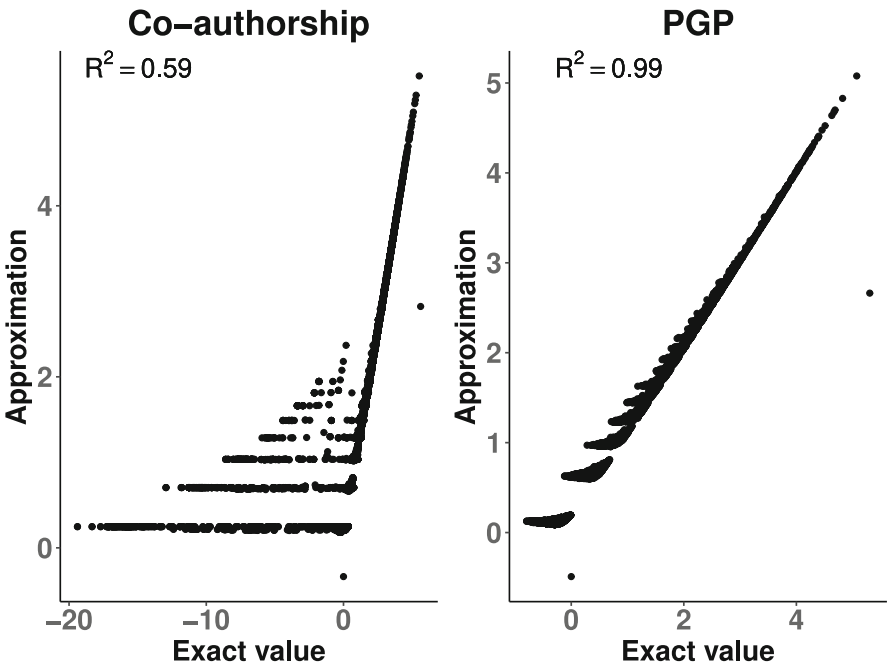

Fig. 5 Scatter plot of the Exact vs. Approximate value of the local measures for the logarithm of the number of spanning trees of two real-world networks

networks, showing a coefficient of determination (R^2) of 1, which indicates perfect agreement between the exact and approximate values. We observed a similar result for the logarithm of the number of spanning trees in the PGP network, with an R^2 of 0.99.

However, there is a noticeable discrepancy in the approximation for the logarithm of the number of spanning trees in the co-authorship network. This discrepancy arises because of the way we computed the exact values. We computed the exact values using the Laplacian matrix's eigenvalue decomposition. Finally, we obtained the local measures from the diagonal elements of $f(\mathbf{M})$. The limitation of this approach is that the Laplacian of the co-authorship network contains several eigenvalues very close to zero, e.g., on the order of 10^{-6}. Calculating the logarithm of such small values results in negative local measures.

5 Conclusion

Here, we introduced a message-passing approach that relies solely on local computations for estimating the trace of matrix functions. Experiments on both simulated and real-world networks demonstrate that our method consistently achieves higher

accuracy than SLQ, which depends on global computations, particularly for configuration and Watts-Strogatz networks, besides maintaining competitive computation times. For Barabási-Albert networks, our approach achieves comparable accuracy to SLQ but with reduced variance, ensuring reliable performance across different network realizations—a key advantage for analyzing networks generated by a preferential attachment mechanism. One disadvantage of our approach is that its computational complexity increases in graphs with highly skewed degree distributions. Recent studies suggest we can address this limitation by incorporating vertex-specific approximations [38]. We also see significant potential in extending our method to solve problems of the form $f(\mathbf{M})\mathbf{b}$, opening the door to efficient computation of centrality measures such as Katz centrality, subgraph communicability, and total communicability [2]. The inherently parallel message-passing algorithm suits it ideally for distributed systems, GPU-based, or graph processor implementations. By leveraging these processors' parallel architecture, we anticipate achieving speed improvements, enabling the analysis of massive networks across a wide range of applications, from social media analytics to biological network simulations.

Appendix A: Walk-summability

A walk w of length $l \geq 0$ in a graph G is a sequence $w = (w_0, w_1, \ldots, w_l)$ of vertices $w_k \in V(G)$ such that $(w_k, w_{k+1}) \in E(G)$. We define the weight of a walk (walk-sum) as the product of edge weights along the walk:

$$\phi(w) = \prod_{k=1}^{l} \mathbf{M}_{w_{k-1} w_k}.$$

Now, let $z \in \{z_1, z_2, \ldots, z_q\}$ be a quadrature point, and $\mathbf{R} = \frac{1}{z}\mathbf{M}$, then the (i, j) element of $\mathbf{R}^l$ can be expressed as a sum of walks $w : i \xrightarrow{l} j$ from i to j of length l:

$$(\mathbf{R}^l)_{ij} = \sum_{v_1 = i, v_2, \ldots, v_l = j} \mathbf{R}_{i,v_1} \mathbf{R}_{v_1,v_2} \ldots \mathbf{R}_{v_{l-1},j} = \sum_{w : i \xrightarrow{l} j} \phi(w).$$

The previous equation holds because only the terms corresponding to walks in the graph have non-zero contributions. The set of walks from i to j of length l is finite; thus, their walk-sum is well-defined. To solve our problem, however, we must account for walk-sums over the set of walks of any length. This might not converge, or convergence may depend on the order of summation. We say that the matrix $\mathbf{R}$ is *walk-summable* (WS) if the sum over all walks w from i to j ($\forall i, j \in V(G)$):

$$\sum_{w : i \to j} \phi(w),$$

converges to the same value for all possible summation orders. From [39], the unordered sum is well-defined if and only if it *converges absolutely*, that is if $\sum_{w:i\to j} |\phi(w)|$ converges. In the following we write $\bar{\mathbf{R}}_{ij} = |\mathbf{R}_{ij}|$.

Proposition 1 (Walk-Summability - WS) *Let $z \in \{z_1, z_2, \ldots, z_q\}$ be a quadrature point, and $\mathbf{R} = \mathbf{M}/z$. Each of the following three conditions is equivalent to the walk-summability of $\mathbf{R}$:*

1. $\sum_{w:i\to j} |\phi(w)|$ converges for all $i, j \in V(G)$.
2. $\sum_{l=0}^{\infty} \bar{\mathbf{R}}^l$ converges.
3. $\rho(\bar{\mathbf{R}}) < 1$.

Proof *(1) $\Rightarrow$ (2).* We examine the convergence of the matrix series in (2) element-wise. Note that $(\bar{\mathbf{R}}^l)_{ij}$ is an absolute walk-sum over all walks of length l from i to j:

$$(\bar{\mathbf{R}}^l)_{ij} = \sum_{w:i \xrightarrow{l} j} |\phi(w)|. \tag{A1}$$

The sum is well-defined because there is a finite number of walks. Since (1) holds, then we can order the sum $\sum_{w:i\to j} |\phi(w)|$ using the absolute convergence property; and by ordering walks by their length, we obtain:

$$\sum_{w:i\to j} |\phi(w)| = \sum_{l} \sum_{w:i \xrightarrow{l} j} |\phi(w)| = \sum_{l} (\bar{\mathbf{R}}^l)_{ij}. \tag{A2}$$

Therefore, the series $\sum_l (\bar{\mathbf{R}}^l)_{ij}$ converges for all i, j.
(2) $\Rightarrow$ (1). The convergence of the sum $\sum_{w:i\to j} |\phi(w)|$ is sufficient to test convergence for any convenient ordering of the walks. Consider the ordering given in (A2), which converges by (2). Therefore, the walk-sums in (1) converge absolutely.
The equivalence *(2) $\Leftrightarrow$ (3)* is proved in [40].

Perron-Frobenius theorem [23]**:** There exists a non-negative eigenvector $\mathbf{x} \geq 0$ of $\bar{\mathbf{R}}$ with eigenvalue $\rho(\bar{\mathbf{R}})$. If the graph G is connected (where $\mathbf{R}_{ij} \neq 0, \forall (i, j) \in E(G)$) then $\rho(\bar{\mathbf{R}})$ and $\mathbf{x}$ are strictly positive and, apart from $\gamma\mathbf{x}$ with $\gamma > 0$, there are no other non-negative eigenvectors of $\bar{\mathbf{R}}$. In addition, we have the following monotonicity property of the spectral radius:

$$\begin{aligned} &(i)\rho(\mathbf{R}) \leq \rho(\bar{\mathbf{R}}), \\ &(ii)\ \text{If } \bar{\mathbf{R}}_1 \leq \bar{\mathbf{R}}_2 \text{ then } \rho(\bar{\mathbf{R}}_1) \leq \rho(\bar{\mathbf{R}}_2). \end{aligned} \tag{A3}$$

Proposition 2 (WS Necessary Conditions) *Let $z \in \{z_1, z_2, \ldots, z_q\}$ be a quadrature point, and $\mathbf{R} = \mathbf{M}/z$. The walk-summability of $\mathbf{R}$ implies the following two equivalent conditions:*

1. $\rho(\mathbf{R}) < 1$,

2. $\sum_{k=0}^{\infty} \mathbf{R}^k = (\mathbf{I} - \mathbf{R})^{-1}$.

Proof WS $\Rightarrow$ (1). WS is equivalent to $\rho(\bar{\mathbf{R}}) < 1$ by Proposition 1. By (A3) we have that $\rho(\mathbf{R}) \leq \rho(\bar{\mathbf{R}})$. Hence, $\rho(\mathbf{R}) < 1$. A proof of the implication (1) $\Rightarrow$ (2) is given in [40].

Therefore, our proposed method works for walk-sum graphs. Furthermore, the message-passing equations will obtain the exact value when the graph does not have primitive cycles longer than $r + 2$.

Appendix B: Detailed performance data

Table 5 Mean signed percent error of the message-passing approach and its 95% confidence interval (shown in square brackets) when obtaining the number of closed walks of length three with different sizes of the neighborhood r and points in the quadrature q

Model	r	Message-passing			
		$q = 8$	$q = 16$	$q = 32$	$q = 64$
Configuration	0	1367.2 [1047;1684]	122.1 [71;166]	-42.7 [-66;-20]	-65.7 [-85;-43]
	1	1431.3 [1100;1755]	195.1 [136;246]	34.8 [13;51]	12.8 [0.1;27]
	2	1485.3 [1144;1821]	229.1 [163;288]	57.6 [27;78]	34.6 [7;55]
Watts-Strogatz	0	-81.7 [-82;-80]	-81.6 [-82;-80]	-81.6 [-82;-80]	-81.6 [-82;-80]
	1	-0.02 [-0.03;-0.01]	0.01 [0.0;0.01]	0.01 [0.0;0.01]	0.01 [0.0;0.01]
	2	4.9 [4;6]	5.0 [4;6]	5.0 [4;6]	5.0 [4;6]
Barabási-Albert	0	269.1 [142;565]	527.4 [399;-713]	364.8 [399;873]	629.4 [405;863]
	1	123.0 [51;383]	192.2 [70;261]	82.6 [-2;165]	5.6 [-6;22]
	2	138.2 [42;480]	335.3 [222;571]	587.4 [484;701]	722.5 [501;928]

We generated 10 graphs with 40 000 vertices using the configuration, Watts-Strogatz, and Barabási-Albert model. The signed percent error is the difference between the exact value and the approximation divided by the exact value. Finally, we multiplied the result by 100

Table 6 Mean signed percent error of the message-passing approach and its 95% confidence interval (between brackets) when obtaining the logarithm of the number of spanning trees with different values of r and q

Model	r	Message-passing			
		$q = 8$	$q = 16$	$q = 32$	$q = 64$
Configuration	0	0.5 [0.5;0.6]	0.2 [0.1;0.2]	0.1 [0.1;0.1]	0.0 [0.0;0.1]
	1	0.5 [0.5;0.6]	0.2 [0.1;0.2]	0.1 [0.0;0.1]	0.0 [0.0;0.1]
	2	0.5 [0.5;0.6]	0.2 [0.1;0.2]	0.1 [0.0;0.1]	0.0 [0.0;0.1]
Watts-Strogatz	0	1.0 [1.0;1.0]	0.8 [0.8;0.8]	0.7 [0.7;0.8]	0.7 [0.7;0.7]
	1	0.5 [0.5;0.6]	0.3 [0.3;0.3]	0.3 [0.2;0.3]	0.2 [0.2;0.3]
	2	0.4 [0.4;0.4]	0.2 [0.2;0.2]	0.1 [0.1;0.1]	0.1 [0.1;0.1]
Barabási-Albert	0	23.7 [21.9;25.1]	13.7 [10.9;15.9]	3.9 [2.8;4.9]	1.0 [0.8;1.3]
	1	23.7 [21.9;25.2]	13.7 [10.9;15.9]	3.9 [2.8;4.9]	1.0 [0.8;1.3]
	2	23.7 [21.9;25.2]	13.7 [10.9;15.9]	3.9 [2.8;4.9]	1.0 [0.8;1.3]

We generated 10 graphs with 40 000 vertices using the configuration, Watts-Strogatz, and Barabási-Albert model. The signed percent error is the difference between the exact value and the approximation divided by the exact value. Finally,and we multiplied the result by 100

Acknowledgements This work has been supported by FAPESP grants 2013/07699-0, 2018/21934-5, 2019/22845-9, and 2020/08343-8, CNPq grant 306811/2022-7, CAPES (finance code 001), Alexander von Humboldt Foundation, the Academy of Medical Sciences - Newton Fund, Wellcome Leap, and the German Academic Exchange Service (DAAD grant no. 57598588).

Author Contributions G.E.C.G. derived the equations and carried out the computational experiments. P.F.S. and A.F. designed and supervised the study. All authors wrote and reviewed the manuscript.

Data Availability Data sharing does not apply to this article as no datasets were generated or analyzed during the current study.

Declarations

Ethics Approval We did not require an ethics approval committee to work on this article.

Competing Interests The authors declare no competing interests.

References

1. Estrada, E., Higham, D.J.: Network properties revealed through matrix functions. SIAM Rev. **52**(4), 696–714 (2010)

2. Benzi, M., Boito, P.: Matrix functions in network analysis. GAMM-Mitteilungen **43**(3), 202000012 (2020)
3. Njotto, L.L., et al.: Centrality measures based on matrix functions. Open J. Discret. Math. **8**(04), 79 (2018)
4. Ubaru, S., Saad, Y.: Applications of trace estimation techniques. In: Kozubek, T., Čermák, M., Tichý, P., Blaheta, R., Šístek, J., Lukáš, D., Jaroš, J. (eds.) High Performance Computing in Science and Engineering, pp. 19–33. Springer, Cham (2018)
5. Rodríguez, J.A., Estrada, E., Gutiérrez, A.: Functional centrality in graphs. Linear Multilinear Algebra **55**(3), 293–302 (2007)
6. Van Mieghem, P.: Graph Spectra for Complex Networks. Cambridge University Press, Shaftesbury Road, Cambridge, CB2 8EA - UK (2010)
7. Zhou, B., Trinajstić, N.: A note on Kirchhoff index. Chem. Phys. Lett. **455**(1–3), 120–123 (2008)
8. Zhou, B., Trinajstić, N.: The Kirchhoff index and the matching number. Int. J. Quantum Chem. **109**(13), 2978–2981 (2009)
9. Milovanović, I., Gutman, I., Milovanović, E.: On Kirchhoff and degree Kirchhoff indices. Filomat. **29**(8), 1869–1877 (2015)
10. Bergermann, K., Stoll, M.: Orientations and matrix function-based centralities in multiplex network analysis of urban public transport. Appl. Netw. Sci. **6**(1), 1–33 (2021)
11. Fuentes, R.D., Donatelli, M., Fenu, C., Mantica, G.: Estimating the trace of matrix functions with application to complex networks. Numer. Algorithms **92**(1), 503–522 (2023)
12. Ubaru, S., Chen, J., Saad, Y.: Fast estimation of $tr(f(A))$ via stochastic Lanczos quadrature. SIAM J. Matrix Anal. Appl. **38**(4), 1075–1099 (2017)
13. Zhang, Y., Leithead, W.E.: Approximate implementation of the logarithm of the matrix determinant in gaussian process regression. J. Stat. Comput. Simul. **77**(4), 329–348 (2007)
14. Hale, N., Higham, N.J., Trefethen, L.N.: Computing A^{α}, $\log(A)$, and related matrix functions by contour integrals. SIAM J. Numer. Anal. **46**(5), 2505–2523 (2008)
15. Han, I., Malioutov, D., Shin, J.: Large-scale log-determinant computation through stochastic Chebyshev expansions. In: International Conference on Machine Learning, pp. 908–917 (2015). PMLR
16. Bickson, D.: Gaussian belief propagation: Theory and aplication. arXiv preprint arXiv:0811.2518 (2008)
17. Ortiz, J., Evans, T., Davison, A.J.: A visual introduction to gaussian belief propagation. arXiv preprint arXiv:2107.02308 (2021)
18. Mézard, M., Montanari, A.: Information, Physics, and Computation. Oxford University Press, North Kettering Business Park, Hipwell Road, Kettering, Northamptonshire. NN14 1UA. (2009)
19. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT press, 314 Main Street, E28. Cambridge, MA 02142 (2009)
20. Cantwell, G.T., Newman, M.E.J.: Message passing on networks with loops. Proc. Natl. Acad. Sci. **116**(47), 23398–23403 (2019)
21. Higham, N.J.: Functions of Matrices: Theory and Computation vol. 104. SIAM, 3600 Market Street, 6th Floor Philadelphia, PA 19104 USA (2008)
22. Guttel, S., Polizzi, E., Tang, P.T.P., Viaud, G.: Zolotarev quadrature rules and load balancing for the FEAST eigensolver. SIAM J. Sci. Comput. **37**(4), 2100–2122 (2015)
23. Malioutov, D.M., Johnson, J.K., Willsky, A.S.: Walk-sums and belief propagation in gaussian graphical models. J. Mach. Learn. Res. **7**, 2031–2064 (2006)
24. Newman, M.: Networks, 780 (2018). https://doi.org/10.1093/oso/9780198805090.001.0001
25. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world'networks. Nature. **393**(6684), 440–442 (1998)
26. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. Science. **286**(5439), 509–512 (1999)
27. Krapivsky, P.L., Redner, S., Leyvraz, F.: Connectivity of growing random networks. Phys. Rev. Lett. **85**(21), 4629 (2000)
28. Acosta, G., Grana, M., Pinasco, J.P.: Condition numbers and scale free graphs. Eur. Phys. J. B-Condensed Matter Complex Syst. **53**, 381–385 (2006)
29. Shao, P.-L.: An improved Lanczos algorithm for solving ill-conditioned linear equations. Comput. Math. Appl. **20**(12), 25–33 (1990)
30. Newman, M.E.J.: The structure of scientific collaboration networks. Proc. Natl. Acad. Sci. U S A. **98**(2), 404–409 (2001)

31. Boguná, M., Pastor-Satorras, R., Díaz-Guilera, A., Arenas, A.: Models of social networks based on social distance attachment. Phys. Rev. E **70**(5), 056122 (2004)
32. Qi, X., Fuller, E., Luo, R., Zhang, C.-q.: A novel centrality method for weighted networks based on the kirchhoff polynomial. Pattern Recognit. Lett. **58**, 51–60 (2015)
33. Harris, J.M., Hirst, J.L., Mossinghoff, M.: Combinatorics and Graph Theory. Springer, New York (2008)
34. Das, K.C., Bapat, R.B.: A sharp upper bound on the spectral radius of weighted graphs. Discret. Math. **308**, 3180–3186 (2008). https://doi.org/10.1016/j.disc.2007.06.020
35. Anderson jr., W.N., Morley, D. Thomas: Eigenvalues of the Laplacian of a graph. Linear Multilinear Algebra. **18**, 141–145 (1985). https://doi.org/10.1080/03081088508817681
36. Abreu, N.M.M.: Old and new results on algebraic connectivity of graphs. Lin. Alg. Appl. **423**, 53–73 (2007)
37. Miller, K.S., Samko, S.G.: Completely monotonic functions. Integr. Transforms Spec. Funct. **12**(4), 389–402 (2001)
38. Cantwell, G.T., Kirkley, A., Radicchi, F.: Heterogeneous message passing for heterogeneous networks. Phys. Rev. E **108**(3), 034310 (2023)
39. Lang, S.: Undergraduate analysis. (2013)
40. Varga, R.S.: Matrix Properties and Concepts, pp. 1–30. Springer, Berlin, Heidelberg (2000). https://doi.org/10.1007/978-3-642-05156-2_1

Authors and Affiliations

Grover Enrique Castro Guzman[1] · Peter Florian Stadler[2,3,4,5,6] · Andre Fujita[1,7]

✉ Andre Fujita
andrefujita@usp.br

Grover Enrique Castro Guzman
grover@usp.br

Peter Florian Stadler
Peter.Stadler@bioinf.uni-leipzig.de

1 Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, Rua do Matão, 1010, São Paulo - SP 05508-090, Brazil

2 Bioinformatics Group, Department of Computer Science; Interdisciplinary Center for Bioinformatics; German Centre for Integrative Biodiversity Research (iDiv) Halle-Jena-Leipzig; Competence Center for Scalable Data Services and Solutions Dresden-Leipzig; Konrad Zuse School of Excellence in Embedded Composite AI Dresden/Leipzig (SECAI), Leipzig University, Härtelstraße 16-18, D-04107 Leipzig, Germany

3 Max Planck Institute for Mathematics in the Sciences, Inselstraße 22, D-04103 Leipzig, Germany

4 Institute for Theoretical Chemistry, University of Vienna, Währingerstraße 17, A-1090 Wien, Austria

5 Facultad de Ciencias, Universidad Nacional de Colombia, Sede Bogotá, Colombia

6 The Santa Fe Institute, 1399 Hyde Park Rd., Santa Fe, NM 87501, USA

7 Division of Network AI Statistics, Medical Institute of Bioregulation, Kyushu University, Fukuoka, Japan