Fast Approximation of Coherence for Second-Order Noisy Consensus Networks

Zuobai Zhang, Wanyue Xu *Student Member, IEEE*, Yuhao Yi, Zhongzhi Zhang, *Member, IEEE*

***Abstract*—It has been recently established that for second-order consensus dynamics with additive noise, the performance measures, including the vertex coherence and network coherence defined, respectively, as the steady-state variance of the deviation of each vertex state from the average and the average steady-state variance of the system, are closely related to the biharmonic distances. However, direct computation of biharmonic distances is computationally infeasible for huge networks with millions of vertices. In this paper, leveraging the implicit fact that both vertex and network coherence can be expressed in terms of the diagonal entries of pseudoinverse $\boldsymbol{L}^{2\dagger}$ of the square of graph Laplacian, we develop a nearly linear time algorithm to approximate all diagonal entries of $\boldsymbol{L}^{2\dagger}$, which has a theoretically guaranteed error for each diagonal entry. The key ingredient of our approximation algorithm is an integration of the Johnson-Lindenstrauss Lemma and Laplacian solvers. Extensive numerical experiments on real-life and model networks are presented, which indicate that our approximation algorithm is both efficient and accurate, and is scalable to large-scale networks with millions of vertices.**

***Index Terms*—Distributed average consensus, multi-agent systems, graph Laplacian, biharmonic distance, Laplacian solver, Gaussian white noise, network coherence**

I. Introduction

A LARGE variety of complex systems display emergent collective behaviors [1], with frequently cited examples including social systems [2], biology systems [3], and multi-agent systems [4]. A lot of collective behaviors can be investigated in the framework of consensus problems, in which all agents reach agreement on a certain quantity or issue, in the scenarios of networks of agents. Consensus dynamics is at the core of many diverse dynamical processes, since it describes various practical problems in both science and engineering areas, such as flocking [5], rendezvous [6], and sensor fusion [7], [8], [9]. In view of the wide applications, the past decade has witnessed a spectacular growth of studies for consensus problems [4], [10], [11], [12].

Many primary aspects of consensus problems in networks have been extensively studied in previous works, including convergence rate [13], [14], [15], robustness to delay [16], [17], [15] and noise [18], [19], [20], [21], [22], [23], [24], [25], [26]. In the context of robustness to noise, the concept of network coherence was introduced by Bamieh and his collaborators to measure the average steady-state variance of node states [22]. For first-order consensus dynamics with stochastic external perturbations, its performance, gauged by network coherence, can be characterized by the resistance distances [27], which are determined by entries of pseudoinverse $\boldsymbol{L}^{\dagger}$ of the graph Laplacian matrix $\boldsymbol{L}$, or the eigenvalues and their corresponding mutually orthogonal unit eigenvectors of $\boldsymbol{L}^{\dagger}$. Thus far, first-order network coherence has been much studied, uncovering the strong connections between structural properties on the scaling of network coherence [18], [19], [20], [21], [22], [23], [24], [25], [26].

Relative to the first-order coherence, there is much less research on second-order coherence [22]. Recently, a unified metric for the performance of second-order noisy consensus has been proposed to measure the variance of the difference between the states of any vertex pair, the variance between a single vertex state and the system average, as well as the total variance of the whole system [28]. This unifying measure is based on biharmonic distance [29], which plays a role similar to the resistance distance for first-order coherence. In addition to measuring the performance of second-order noisy consensus dynamics, the biharmonic distance has also been successfully applied to different areas, for example, computer graphics [29]. However, exact straightforward computation of biharmonic distance involves inverting a matrix, which is computationally challenging task for large networks. It is thus of great interest to develop a fast algorithm for computing guaranteed biharmonic distances or their aggregates for relevant practical applications, such as second-order coherence.

In this paper, we propose a randomized approximation algorithm to evaluate the coherence for every node and for the whole system in second-order consensus networks with additive noise. By making use of the implicit fact that second-order coherence can be expressed in terms of the diagonal elements of the pseudoinverse $\boldsymbol{L}^{2\dagger}$ for the square of Laplacian matrix $\boldsymbol{L}$ [28], we reduce the computation problem to approximating the diagonal entries of $\boldsymbol{L}^{2\dagger}$. Our algorithm is based on the Johnson-Lindenstrauss Lemma [30], [31], [32] and Laplacian solvers [33], [34], [35], [36], [37]. It returns an approximation for every diagonal entries of $\boldsymbol{L}^{2\dagger}$ in nearly linear time with respect to the number of edges. In addition, our algorithm has a provable approximation error with high probability. Finally, we execute numerical experiments on a large variety of real and model networks, the results of which demonstrate that our

This work was supported in part by the Shanghai Municipal Science and Technology Major Project (No. 2018SHZDZX01) and ZJLab, the National Natural Science Foundation of China (Nos. 61872093, 61803248), and the National Key R & D Program of China (No. 2018YFB1305104). Zuobai Zhang was also supported by Fudan¡s Undergraduate Research Opportunities Program (FDUROP) under Grant No. 19914. *(Corresponding author: Zhongzhi Zhang.)*

Zuobai Zhang, Wanyue Xu, Yuhao Yi, and Zhongzhi Zhang are with the Shanghai Key Laboratory of Intelligent Information Processing, School of Computer Science, Fudan University, Shanghai, 200433, China; and are also with Fudan-Zhongan Joint Laboratory of Blockchain and Information Security, Shanghai Engineering Research Institute of Blockchain, Fudan University, Shanghai 200433, China. (e-mail: 17300240035@fudan.edu.cn; xuwy@fudan.edu.cn; yhyi15@fudan.edu.cn; zhangzz@fudan.edu.cn).

Manuscript received **** **, 2019; revised **** **, 2019.

approximation algorithm is both efficient and accurate.

II. Preliminary

In this section, we give a brief introduction to some essential concepts about graphs, related matrices, graph Laplacian and its pseudoinverse, resistance distance, biharmonic distance, and second-order noisy consensus dynamics.

A. Graph and Related Matrices

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a connected undirected binary graph with vertex/node set $\mathcal{V}$ and edge/link set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, where the number of vertices is $n = |\mathcal{V}|$ and the number of edges is $m = |\mathcal{E}|$. We use $u \sim v$ to indicate that a pair of vertices u and v are directly linked to each other by an edge.

For a graph $\mathcal{G}$, many of its properties are encoded in its adjacency matrix $\boldsymbol{A}$, whose entry a_{ij} indicates the adjacency relation between vertices i and j: $a_{ij} = 1$ if $i \sim j \in \mathcal{E}$ and $a_{ij} = 0$ otherwise. The degree of vertex i is $d_i = \sum_{i=1}^{n} a_{ij}$. Let $\boldsymbol{D}$ denote the diagonal degree matrix, where the ith diagonal entry is d_i, while all other entries are zeros. Then, the Laplacian matrix $\boldsymbol{L}$ of $\mathcal{G}$ is defined to be $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$. Let $\mathbf{1}$ and $\boldsymbol{J}$ denote, respectively, the vector and the matrix of appropriate dimensions with all entries being ones. Then, $\boldsymbol{J} = \mathbf{1}\mathbf{1}^\top$. Let $\mathbf{0}$ and $\mathbf{O}$ denote, respectively, the zero vector and zero matrix of appropriate dimensions. Since $\boldsymbol{D}\mathbf{1} = \boldsymbol{A}\mathbf{1}$, then $\boldsymbol{L}\mathbf{1} = \mathbf{0}$ and $\boldsymbol{L}\boldsymbol{J} = \boldsymbol{J}\boldsymbol{L} = \mathbf{O}$.

Let $\boldsymbol{B} \in \mathbb{R}^{m\times n}$ be the edge-vertex incidence matrix of graph $\mathcal{G}$. For every edge e with vertices i and j, a direction is assigned arbitrarily. We use $\boldsymbol{b}_e^\top$ to denote the row of matrix $\boldsymbol{B}$ corresponding to edge e. Then the entry b_{eu} at row associated with edge e and column corresponding to vertex u is: $b_{eu} = 1$ if vertex u is the tail of edge e, $b_{eu} = -1$ if vertex u is the head of edge e, and $b_{eu} = 0$ otherwise. Let $\boldsymbol{e}_u$ be the u-th canonical basis of the space $\mathbb{R}^n$. Then for an edge e linking two vertices i and j, $\boldsymbol{b}_e$ can also be represented as $\boldsymbol{b}_e = \boldsymbol{e}_i - \boldsymbol{e}_j$, and the Laplacian matrix $\boldsymbol{L}$ of $\mathcal{G}$ can be recast as $\boldsymbol{L} = \boldsymbol{B}^\top\boldsymbol{B} = \sum_{e\in\mathcal{E}} \boldsymbol{b}_e\boldsymbol{b}_e^\top$.

For a matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ with entries x_{ij}, $i = 1, 2, \cdots, m$ and $j = 1, 2, \cdots, n$, its Frobenius norm $\|\boldsymbol{X}\|_F$ is defined as

$$\|\boldsymbol{X}\|_F := \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2} = \sqrt{\mathrm{Tr}\left(\boldsymbol{X}^\top\boldsymbol{X}\right)}.$$

B. Spectrum of Graph Laplacian and its Pseudoinverse

For a connected undirected unweighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, its Laplacian matrix $\boldsymbol{L}$ is symmetric and positive semidefinite. Then, all the eigenvalues of $\boldsymbol{L}$ are non-negative, with a unique zero eigenvalue. Let $0 = \lambda_1 < \lambda_2 \le \lambda_3 \le \cdots \le \lambda_{n-1} \le \lambda_n$ denote the n eigenvalues of matrix $\boldsymbol{L}$, and let $\boldsymbol{u}_k$, $k = 1, 2, \ldots, n$, denote their corresponding mutually orthogonal unit eigenvectors. Then, $\boldsymbol{L}$ has a spectral decomposition of the form $\boldsymbol{L} = \sum_{k=1}^{n} \lambda_k \boldsymbol{u}_k\boldsymbol{u}_k^\top$. Thus, the entry $\boldsymbol{L}_{ij}$ at row i and column j of $\boldsymbol{L}$ can be expressed as $\boldsymbol{L}_{ij} = \sum_{k=2}^{n} \lambda_k u_{ki}u_{kj}$, where u_{ki} is the i-th component of vector $\boldsymbol{u}_k$. It has been shown [38] that among all n-node connected undirected unweighted graphs, the largest eigenvalue λ_n of Laplacian matrix $\boldsymbol{L}$ satisfies $\lambda_n \le n$, with equality if and only if the largest degree of vertices in $\mathcal{G}$ is $n - 1$.

The Laplacian matrix $\boldsymbol{L}$ is singular and cannot be inverted, since 0 is one of its eigenvalues. As a substitute for the inverse, we use the Moore-Penrose generalized inverse of $\boldsymbol{L}$, which we simply call pseudoinverse of $\boldsymbol{L}$ [39]. As customary, we use $\boldsymbol{L}^\dagger$ to denote its pseudoinverse, which can be written as

$$\boldsymbol{L}^\dagger = \sum_{k=2}^{n} \frac{1}{\lambda_k}\boldsymbol{u}_k\boldsymbol{u}_k^\top. \tag{1}$$

Then, the ij-th entry $\boldsymbol{L}_{ij}^\dagger$ of $\boldsymbol{L}^\dagger$ can be represented as $\boldsymbol{L}_{ij}^\dagger = \sum_{k=2}^{n} \frac{1}{\lambda_k} u_{ki}u_{kj}$, and the ith diagonal entry $\boldsymbol{L}_{ii}^\dagger$ of $\boldsymbol{L}^\dagger$ is $\sum_{k=2}^{n} \frac{1}{\lambda_k} u_{ki}^2$. Since $\boldsymbol{L}$ is symmetric, so it is with $\boldsymbol{L}^\dagger$ [40].

The symmetry of $\boldsymbol{L}$ and $\boldsymbol{L}^\dagger$ implies that they share the same null space [39]. Since $\boldsymbol{L}\mathbf{1} = \mathbf{0}$, then $\boldsymbol{L}^\dagger\mathbf{1} = \mathbf{0}$. Furthermore, by using $\boldsymbol{J} = \mathbf{1}\mathbf{1}^\top$, we have $\boldsymbol{L}\boldsymbol{J} = \boldsymbol{J}\boldsymbol{L} = \boldsymbol{L}^\dagger\boldsymbol{J} = \boldsymbol{J}\boldsymbol{L}^\dagger = \mathbf{O}$. Let $\boldsymbol{I}$ be the identity matrix of an appropriate dimension. Using the the above spectral decompositions of $\boldsymbol{L}$ and $\boldsymbol{L}^\dagger$, we have

$$\left(\boldsymbol{L} + \frac{1}{n}\boldsymbol{J}\right)\left(\boldsymbol{L}^\dagger + \frac{1}{n}\boldsymbol{J}\right) = \boldsymbol{I}.$$

Then, the pseudoinverse $\boldsymbol{L}^\dagger$ of $\boldsymbol{L}$ can be represented as

$$\boldsymbol{L}^\dagger = \left(\boldsymbol{L} + \frac{1}{n}\boldsymbol{J}\right)^{-1} - \frac{1}{n}\boldsymbol{J}. \tag{2}$$

This expression was explicitly stated in [41] and was implicitly applied in [42], [43].

C. Square of Graph Laplacian and Its Pseudoinverse

In addition to graph Laplacian $\boldsymbol{L}$ itself, its square $\boldsymbol{L}^2$ also received much attention, since it appears in many practical scenarios [29], [44]. By definition, the eigenvalues and the corresponding eigenvectors of $\boldsymbol{L}^2$ are, respectively, λ_k^2 and $\boldsymbol{u}_k$, $k = 1, 2, \cdots, n$. Thus, $\boldsymbol{L}^2$ has the following spectral decomposition: $\boldsymbol{L}^2 = \sum_{k=1}^{n} \lambda_k^2 \boldsymbol{u}_k\boldsymbol{u}_k^\top$, which indicates that the entry $\boldsymbol{L}_{ij}^2$ at row i and column j of $\boldsymbol{L}^2$ is $\boldsymbol{L}_{ij}^2 = \sum_{k=2}^{n} \lambda_k^2 u_{ki}u_{kj}$. Matrix $\boldsymbol{L}^2$ is also not invertibale, and we write its pseudoinverse as $\boldsymbol{L}^{2\dagger}$ which satisfies

$$\boldsymbol{L}^{2\dagger} = \sum_{k=2}^{n} \frac{1}{\lambda_k^2}\boldsymbol{u}_k\boldsymbol{u}_k^\top = (\boldsymbol{L}^\dagger)^2, \tag{3}$$

implying that the ij-th entry of $\boldsymbol{L}^{2\dagger}$ is $\boldsymbol{L}_{ij}^\dagger = \sum_{k=2}^{n} \frac{1}{\lambda_k^2} u_{ki}u_{kj}$, and particularly, the ith diagonal entry of $\boldsymbol{L}^{2\dagger}$ is $\boldsymbol{L}_{ii}^{2\dagger} = \sum_{k=2}^{n} \frac{1}{\lambda_k^2} u_{ki}^2$.

Similar to $\boldsymbol{L}$ and its pseudoinverse $\boldsymbol{L}^\dagger$, for $\boldsymbol{L}^2$ and $\boldsymbol{L}^{2\dagger}$, we have $\boldsymbol{L}^2\mathbf{1} = \mathbf{0}$, $\boldsymbol{L}^{2\dagger}\mathbf{1} = \mathbf{0}$, $\boldsymbol{L}^2\boldsymbol{J} = \boldsymbol{J}\boldsymbol{L}^2 = \boldsymbol{L}^{2\dagger}\boldsymbol{J} = \boldsymbol{J}\boldsymbol{L}^{2\dagger} = \mathbf{O}$, and $\left(\boldsymbol{L}^2 + \frac{1}{n}\boldsymbol{J}\right)\left(\boldsymbol{L}^{2\dagger} + \frac{1}{n}\boldsymbol{J}\right) = \boldsymbol{I}$ that leads to the following expression of the pseudoinverse $\boldsymbol{L}^{2\dagger}$ for $\boldsymbol{L}^2$:

$$\boldsymbol{L}^{2\dagger} = \left(\boldsymbol{L}^2 + \frac{1}{n}\boldsymbol{J}\right)^{-1} - \frac{1}{n}\boldsymbol{J}. \tag{4}$$

The above properties and expression for $\boldsymbol{L}^{2\dagger}$ are useful for the approximation analysis of our algorithm that will be presented in Section III.

D. Resistance Distance

For an arbitrary undirected unweightd graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the resistance distance [27] $d_R(u, v)$ between two nodes u and v is defined as the effective resistance between u and v in the corresponding electrical network [45] obtained from $\mathcal{G}$ by considering every edge in $\mathcal{E}$ as a unit resistor and every node in $\mathcal{V}$ as an junction between resistors. It was shown [27] that $d_R(u, v)$ can be expressed in terms of the entries of matrix $\boldsymbol{L}^\dagger$ as $d_R(u, v) = \boldsymbol{L}^\dagger_{uu} + \boldsymbol{L}^\dagger_{vv} - 2\boldsymbol{L}^\dagger_{uv}$.

For a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the sum of resistance distances over all its node pairs is called the *Kirchhoff index* [27] and is denoted by $R(\mathcal{G})$. That is,

$$R(\mathcal{G}) = \sum_{\substack{u,v\in\mathcal{V}\\ u<v}} d_R(u, v).$$

It is has been shown that $R(\mathcal{G}) = n\mathrm{Tr}(\boldsymbol{L}^\dagger)$ [46] and the maximum value of $R(\mathcal{G})$ among all n-node graphs is equal to $n(n^2-1)/6$, which can be achieved when $\mathcal{G}$ is the n-node path graph [47]. This maximum value for Kirchhoff index will be used in the proof of Lemma 3.1 to provide a guaranteed theoretical error of our algorithm.

E. Biharmonic Distance

As the case of Laplacian matrix $\boldsymbol{L}$, the entries of pseudoinverse for its square $\boldsymbol{L}^2$ can also be used to define distance metrics between vertices in a graph, such as biharmonic distance [29], [48].

Definition 2.1: Let $\mathcal{G}$ be an undirected connected graph. The biharmonic distance $d_B(u, v)$ between two vertices u and v in $\mathcal{G}$ is defined by

$$d^2_B(u, v) = \boldsymbol{L}^{2\dagger}_{uu} + \boldsymbol{L}^{2\dagger}_{vv} - 2\boldsymbol{L}^{2\dagger}_{uv}\,. \tag{5}$$

The biharmonic distance has some advantages over the resistance distance and geodesic distance in some realistic contexts, e.g., computer graphics [29]. Based on biharmonic distance, one can also define the following graph index [28].

Definition 2.2: For a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the *biharmonic Kirchhoff index* $D^2_B(\mathcal{G})$ of the whole graph is

$$D^2_B(\mathcal{G}) = \sum_{\substack{u,v\in\mathcal{V}\\ u<v}} d^2_B(u, v)\,. \tag{6}$$

According to (5), one obtains

$$D^2_B(\mathcal{G}) = n\sum_{v=1}^{n} \boldsymbol{L}^{2\dagger}_{vv} = n\,\mathrm{Tr}(\boldsymbol{L}^{2\dagger})\,. \tag{7}$$

The biharmonic Kirchhoff index was previously used to measure the robustness of the second-order noisy consensus problem without leaders [22].

F. Second-Order Noisy Consensus Dynamics

In the second-order consensus problem, each vertex $u \in \mathcal{V}$ in the network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ has two scalar-valued states $x_{1u}(t)$ and $x_{2u}(t)$ at any time t. The states of all vertices can be represented by two vectors, $\boldsymbol{x}_1(t)$ and $\boldsymbol{x}_2(t)$, where $\boldsymbol{x}_1(t)$ is the position vector, while $\boldsymbol{x}_2(t)$ is the velocity vector, with $\boldsymbol{x}_2(t)$ being the first derivative of $\boldsymbol{x}_1(t)$ with respect to t. Every vertex v updates its state by setting $\dot{x}_{2u}(t)$ according to the differences of its state, $x_{1u}(t)$ and $x_{2u}(t)$, and the states of its neighbors. The following equation describes the second-order consensus algorithm with Gaussian white noise:

$$\begin{bmatrix} \dot{\boldsymbol{x}}_1(t) \\ \dot{\boldsymbol{x}}_2(t) \end{bmatrix} = \begin{bmatrix} \mathbf{O} & \boldsymbol{I} \\ -\alpha\boldsymbol{L} & -\beta\boldsymbol{L} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1(t) \\ \boldsymbol{x}_2(t) \end{bmatrix} + \begin{bmatrix} \mathbf{O} \\ \boldsymbol{I} \end{bmatrix} \boldsymbol{w}(t)\,, \tag{8}$$

where α and β are, respectively, positive constant system gains on $\boldsymbol{x}_1(t)$ and $\boldsymbol{x}_2(t)$, and $\boldsymbol{w}(t)$ is an n-dimensional vector of uncorrelated Gaussian white noise processes imposed on state $\boldsymbol{x}_2(t)$. In this paper, without loss of generality, we suppose $\alpha = 1$ and $\beta = 1$.

Since the state $x_{2u}(t)$ of each vertex u is subjected to additive noise, the system cannot reach exact consensus. Thus, we are concerned with the deviations of the position states of the vertices from consensus. Concretely, we focus on two performance measures related to these deviations [28]. The first one is the variance of the difference between the state of a vertex and the current average value of states for all vertices, while the second one is the average of the first one over all nodes. Let $\bar{x}_1(t)$ denote the average state at time t, namely, $\bar{x}_1(t) = \frac{1}{n}\mathbf{1}^\top \boldsymbol{x}_1(t)$.

Definition 2.3: For a vertex $j \in \mathcal{V}$ in graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the *vertex coherence* $H_{\mathrm{SO}}(u)$ is the steady-state variance of the difference between $x_{1u}(t)$ and $\bar{x}_1(t)$, i.e.,

$$H_{\mathrm{SO}}(u) = \lim_{t\to\infty} \mathbb{E}[(x_{1u}(t) - \bar{x}_1(t))^2]\,. \tag{9}$$

The second performance measure we are interested in is the average variance of the whole system.

Definition 2.4: For a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with n vertices, its *average coherence* $H_{\mathrm{SO}}(\mathcal{G})$ is defined as the average of coherence for all vertices in $\mathcal{V}$, i.e.,

$$H_{\mathrm{SO}}(\mathcal{G}) = \frac{1}{n}\lim_{t\to\infty} \sum_{u\in\mathcal{V}} \mathbb{E}[(x_{1u}(t) - \bar{x}_1(t))^2]\,. \tag{10}$$

It has been established implicitly [28] that for a graph $\mathcal{G}$ both $H_{\mathrm{SO}}(u)$ and $H_{\mathrm{SO}}(\mathcal{G})$ are related to biharmonic distances via the diagonal entries of $\boldsymbol{L}^{2\dagger}$:

$$H_{\mathrm{SO}}(u) = \frac{1}{2}\boldsymbol{L}^{2\dagger}_{uu}\,. \tag{11}$$

and

$$H_{\mathrm{SO}}(\mathcal{G}) = \frac{1}{2n}\sum_{k=2}^{n}\frac{1}{\lambda_k^2} = \frac{1}{2n}\sum_{u=1}^{n}\boldsymbol{L}^{2\dagger}_{uu}\,. \tag{12}$$

Equations (11) and (12) reduce the problems of determining second-order vertex coherence $H_{\mathrm{SO}}(u)$ and network coherence $H_{\mathrm{SO}}(\mathcal{G})$ to evaluating the diagonal entries of matrix $\boldsymbol{L}^{2\dagger}$.

G. Johnson-Lindenstrauss Lemma and Laplacian solvers

There are two key ingredients of our algorithm, both of which play an important role in the approximation analysis of our algorithm. One ingredient is the Johnson-Lindenstrauss Lemma [30], [31], [32], given by the following lemma.

Lemma 2.5: Given fixed vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n \in \mathbb{R}^d$ and $\epsilon > 0$, let $\boldsymbol{Q}_{k\times d}$, $k \geq 24\log n/\epsilon^2$ be a matrix with each entry

equal to $1/\sqrt{k}$ or $-1/\sqrt{k}$ with the same probability $1/2$. Then with probability at least $1-1/n$,

$$(1-\epsilon)\|\boldsymbol{v}_i-\boldsymbol{v}_j\|^2 \le \|\boldsymbol{Q}\boldsymbol{v}_i-\boldsymbol{Q}\boldsymbol{v}_j\|^2 \le (1+\epsilon)\|\boldsymbol{v}_i-\boldsymbol{v}_j\|^2$$

for all pairs $i,j \le n$.

The other ingredient is the Laplacian solvers [33], [34], [35], [36], [37]. We here apply the nearly-linear time solver from [37], whose performance is characterized in the following lemma. In the sequel, for the convenience of description, we use the notation $\widetilde{O}(\cdot)$ to hide $\mathrm{poly}\log$ factors.

Lemma 2.6: There is an algorithm $\boldsymbol{x} = \texttt{LaplSolve}(\boldsymbol{L}, \boldsymbol{z}, \delta)$ which takes a Laplacian matrix $\boldsymbol{L}$, a column vector $\boldsymbol{z}$, and an error parameter $\delta > 0$, and returns a column vector $\boldsymbol{x}$ satisfying $\mathbf{1}^\top \boldsymbol{x} = 0$ and

$$\|\boldsymbol{x}-\boldsymbol{L}^\dagger \boldsymbol{z}\|_{\boldsymbol{L}} \le \delta \|\boldsymbol{L}^\dagger \boldsymbol{z}\|_{\boldsymbol{L}},$$

where $\|\boldsymbol{z}\|_{\boldsymbol{L}} = \sqrt{\boldsymbol{z}^\top \boldsymbol{L}\boldsymbol{z}}$. The algorithm runs in expected time $\widetilde{O}\left(m\log(1/\delta)\right)$.

III. Fast Algorithm for Estimating Coherence of Second-Order Noisy Networks

Equations (4), (11) and (12) show that direct exact evaluation of vertex coherence $H_{\mathrm{SO}}(u)$ and network coherence $H_{\mathrm{SO}}(\mathcal{G})$ needs to compute the diagonal entries of matrix $\boldsymbol{L}^{2\dagger}$, which runs in $O(n^3)$ time. To avoid the time-consuming matrix inverse operation, in this section, we develop a fast randomized algorithm, which approximates vertex coherence $H_{\mathrm{SO}}(u)$ of every vertex u and network coherence $H_{\mathrm{SO}}(\mathcal{G})$ for an arbitrary graph $\mathcal{G}$. Our algorithm has a nearly-linear time and space cost with respect to the number of edges. Furthermore, it has a guaranteed theoretical error with high probability.

Before introducing our algorithm, we first rewrite the diagonal entry of $\boldsymbol{L}^{2\dagger}$ corresponding to vertex u in terms of an Euclidian norm as $\boldsymbol{L}^{2\dagger}_{uu} = \boldsymbol{e}_u^\top \boldsymbol{L}^{2\dagger}\boldsymbol{e}_u = \|\boldsymbol{L}^\dagger \boldsymbol{e}_u\|^2$. Thus, the computation of $\boldsymbol{L}^{2\dagger}_{uu}$ is reduced to determining the ℓ_2 norm of a vector in space $\mathbb{R}^n$. However, exactly computing $\boldsymbol{L}^{2\dagger}_{uu}$ by evaluating an ℓ_2 norm still has a high time complexity. In what follows, instead of exactly calculating $\|\boldsymbol{L}^\dagger \boldsymbol{e}_u\|^2$, we will present an algorithm for approximately estimating it, which remarkably reduce the computational cost. Lemma 2.5 shows that the ℓ_2 norm $\|\boldsymbol{L}^\dagger \boldsymbol{e}_u\|^2$ can be nearly preserved by projecting the n-dimensional vector $\boldsymbol{L}^\dagger \boldsymbol{e}_u$, $u \in \mathcal{V}$, onto a $O(\log n)$-dimensional subspaces. Let $\boldsymbol{Q}$ be a $k \times n$ random projection matrix with each entry being $1/\sqrt{k}$ or $-1/\sqrt{k}$ with identical probability. Then $\|\boldsymbol{L}^\dagger \boldsymbol{e}_u\|$ can be well approximated by $\|\boldsymbol{Q}\boldsymbol{L}^\dagger \boldsymbol{e}_u\|$.

As indicated in (2), directly computing $\boldsymbol{L}^\dagger$ involves inverting matrix $\boldsymbol{L} + \frac{1}{n}\boldsymbol{J}$, which is still intractable for huge networks. To overcome the difficulty, we resort to the nearly-linear time Laplacian solver [37] stated in Lemma 2.6. According to Lemma 2.6, in order to compute $\boldsymbol{Y} = \boldsymbol{Q}\boldsymbol{L}^\dagger$ without inverting matrix $\boldsymbol{L} + \frac{1}{n}\boldsymbol{J}$, we can alternatively solve the system of equations $\boldsymbol{L}\boldsymbol{y}_i = \boldsymbol{q}_i$ obeying $\mathbf{1}^\top \boldsymbol{y}_i = 0$ for $i = 1, \ldots, k$, where $\boldsymbol{y}_i^\top$ and $\boldsymbol{q}_i^\top$ are the ith row of $\boldsymbol{Y}$ and $\boldsymbol{Q}$, respectively. The reason can be explained as follows. Let $\boldsymbol{z}_i$ be any solution of $\boldsymbol{L}\boldsymbol{y}_i = \boldsymbol{q}_i$, satisfying $\mathbf{1}^\top \boldsymbol{z}_i = 0$. We now show that $\boldsymbol{z}_i = \boldsymbol{L}^\dagger \boldsymbol{q}_i$. For this purpose, we evaluate $\boldsymbol{L}^\dagger \boldsymbol{L}(\boldsymbol{z}_i - \boldsymbol{L}^\dagger \boldsymbol{q}_i)$ as

$$\boldsymbol{L}^\dagger \boldsymbol{L}(\boldsymbol{z}_i - \boldsymbol{L}^\dagger \boldsymbol{q}_i) = \boldsymbol{L}^\dagger(\boldsymbol{L}\boldsymbol{z}_i) - \boldsymbol{L}^\dagger \boldsymbol{L}\boldsymbol{L}^\dagger \boldsymbol{q}_i = \boldsymbol{L}^\dagger \boldsymbol{q}_i - \boldsymbol{L}^\dagger \boldsymbol{q}_i = 0.$$

Since the null space of $\boldsymbol{L}^\dagger \boldsymbol{L}$ is $\mathbf{1}$, we obtain $\boldsymbol{z}_i - \boldsymbol{L}^\dagger \boldsymbol{q}_i = c\mathbf{1}$, where c is an arbitrary real number. By left multiplying $\mathbf{1}^\top$ on both sides of $\boldsymbol{z}_i - \boldsymbol{L}^\dagger \boldsymbol{q}_i = c\mathbf{1}$ leads to $c = 0$. Then, $\boldsymbol{z}_i = \boldsymbol{L}^\dagger \boldsymbol{q}_i$. Therefore, by Lemma 2.6 the solution $\boldsymbol{x}$ returned by Laplacian solver $\texttt{LaplSolve}(\boldsymbol{L}, \boldsymbol{q}_i, \delta)$ is a good approximation of $\boldsymbol{y}_i$.

Both Lemma 2.5 and Lemma 2.6 play an important role in the proof of the theoretical error bound of our algorithm, which is stated in the following lemma.

Lemma 3.1: Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with n vertices, an approximate factor $0 < \epsilon \le 1/2$, and a $k \times n$ matrix $\boldsymbol{Y}$ that satisfy

$$(1-\epsilon)\boldsymbol{L}^{2\dagger}_{uu} \le \|\boldsymbol{Y}\boldsymbol{e}_u\|^2 \le (1+\epsilon)\boldsymbol{L}^{2\dagger}_{uu},$$

for any node $u \in V$ and

$$(1-\epsilon)\|\boldsymbol{L}^\dagger(\boldsymbol{e}_u - \boldsymbol{e}_v)\|^2 \le \|\boldsymbol{Y}(\boldsymbol{e}_u - \boldsymbol{e}_v)\|^2 \le (1+\epsilon)\|\boldsymbol{L}^\dagger(\boldsymbol{e}_u - \boldsymbol{e}_v)\|^2$$

for any pair of nodes u and v with $u, v \in \mathcal{V}$, let $\boldsymbol{y}_i$ be the i-th row of $\boldsymbol{Y}$ and let $\tilde{\boldsymbol{y}}_i$ be an approximation of $\boldsymbol{y}_i$ for all $i \in \{1, 2, ..., k\}$, satisfying

$$\|\boldsymbol{y}_i - \tilde{\boldsymbol{y}}_i\|_{\boldsymbol{L}} \le \delta \|\boldsymbol{y}_i\|_{\boldsymbol{L}}, \tag{13}$$

where

$$\delta \le \frac{\epsilon}{3}\sqrt{\frac{6(n-1)(1-\epsilon)}{n^5(n+1)(1+\epsilon)}}. \tag{14}$$

Then for an arbitrary node $u \in \mathcal{V}$,

$$(1-\epsilon)^2 \boldsymbol{L}^{2\dagger}_{uu} \le \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|^2 \le (1+\epsilon)^2 \boldsymbol{L}^{2\dagger}_{uu}, \tag{15}$$

where $\tilde{\boldsymbol{Y}}^\top = [\tilde{\boldsymbol{y}}_1, \tilde{\boldsymbol{y}}_2, ..., \tilde{\boldsymbol{y}}_k]$.

Proof. In order to prove (15), it is sufficient to show that for an arbitrary node $u \in \mathcal{V}$,

$$\begin{aligned}
&\left|\|\boldsymbol{Y}\boldsymbol{e}_u\|^2 - \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|^2\right| \\
&= \left|\|\boldsymbol{Y}\boldsymbol{e}_u\| - \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|\right| \times \left|\|\boldsymbol{Y}\boldsymbol{e}_u\| + \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|\right| \\
&\le \left(\frac{2\epsilon}{3} + \frac{\epsilon^2}{9}\right)\|\boldsymbol{Y}\boldsymbol{e}_u\|^2,
\end{aligned} \tag{16}$$

which is satisfied if

$$\left|\|\boldsymbol{Y}\boldsymbol{e}_u\| - \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|\right| \le \frac{\epsilon}{3}\|\boldsymbol{Y}\boldsymbol{e}_u\|. \tag{17}$$

This can be accounted for by the following arguments. First, if $\left|\|\boldsymbol{Y}\boldsymbol{e}_u\|^2 - \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|^2\right| \le \left(\frac{2\epsilon}{3} + \frac{\epsilon^2}{9}\right)\|\boldsymbol{Y}\boldsymbol{e}_u\|^2$, we have

$$\begin{aligned}
\left(1 - \frac{2\epsilon}{3} - \frac{\epsilon^2}{9}\right)\|\boldsymbol{Y}\boldsymbol{e}_u\|^2 &\le \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|^2 \\
&\le \left(1 + \frac{2\epsilon}{3} + \frac{\epsilon^2}{9}\right)\|\boldsymbol{Y}\boldsymbol{e}_u\|^2,
\end{aligned}$$

which, together with $0 < \epsilon \le 1/2$ (implying $\epsilon^2/9 \le \epsilon/3$) and the hypothesis that $(1-\epsilon)\boldsymbol{L}^{2\dagger}_{uu} \le \|\boldsymbol{Y}\boldsymbol{e}_u\|^2 \le (1+\epsilon)\boldsymbol{L}^{2\dagger}_{uu}$,

yields (15). In turn, if (17) holds true, then $\|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\| \le (1+\frac{\epsilon}{3})\|\boldsymbol{Y}\boldsymbol{e}_u\|$. Hence,

$$\left|\|\boldsymbol{Y}\boldsymbol{e}_u\| + \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|\right| \le \left(2+\frac{\epsilon}{3}\right)\|\boldsymbol{Y}\boldsymbol{e}_u\|,$$

which leads to (16).

We next prove that (17) holds true. By applying the triangle inequality twice, we obtain

$$\begin{aligned}
&\left|\|\boldsymbol{Y}\boldsymbol{e}_u\| - \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|\right| \le \|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})\boldsymbol{e}_u\| \\
=&\left\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})\left(\boldsymbol{e}_u - \frac{1}{n}\mathbf{1}\right)\right\| \\
=&\frac{1}{n}\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})\sum_{v\neq u}(\boldsymbol{e}_u-\boldsymbol{e}_v)\| \\
\le&\frac{1}{n}\sum_{v\neq u}\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})(\boldsymbol{e}_u-\boldsymbol{e}_v)\|.
\end{aligned}$$

Let P_{uv} be a simple path connecting vertices u and v. Again, using the triangle inequality, we obtain

$$\begin{aligned}
&\frac{1}{n}\sum_{v\neq u}\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})(\boldsymbol{e}_u-\boldsymbol{e}_v)\| \\
\le&\frac{1}{n}\sum_{v\neq u}\sum_{(a,b)\in P_{uv}}\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})(\boldsymbol{e}_a-\boldsymbol{e}_b)\| \\
\le&\Big(\sum_{v\neq u}\sum_{(a,b)\in P_{uv}}\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})(\boldsymbol{e}_a-\boldsymbol{e}_b)\|^2\Big)^{1/2} \\
\le&n^{1/2}\Big(\sum_{(a,b)\in\mathcal{E}}\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})(\boldsymbol{e}_a-\boldsymbol{e}_b)\|^2\Big)^{1/2} \\
=&n^{1/2}\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})\boldsymbol{B}^\top\|_F.
\end{aligned}$$

Note that the second inequality can be derived by Cauchy-Schwarz Inequality.

We now transform the above-obtained Frobenius norm $n^{1/2}\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})\boldsymbol{B}^\top\|_F$ into the $\boldsymbol{L}$-norm as

$$\begin{aligned}
&n^{1/2}\|(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})\boldsymbol{B}^\top\|_F \\
=&n^{1/2}\sqrt{\mathrm{Tr}\left((\boldsymbol{Y}-\tilde{\boldsymbol{Y}})\boldsymbol{B}^\top\boldsymbol{B}(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})^\top\right)} \\
=&n^{1/2}\sqrt{\mathrm{Tr}\left((\boldsymbol{Y}-\tilde{\boldsymbol{Y}})\boldsymbol{L}(\boldsymbol{Y}-\tilde{\boldsymbol{Y}})^\top\right)} \\
=&n^{1/2}\sqrt{\sum_{i=1}^{k}(\boldsymbol{y}_i-\tilde{\boldsymbol{y}}_i)^\top\boldsymbol{L}(\boldsymbol{y}_i-\tilde{\boldsymbol{y}}_i)} \\
\le&n^{1/2}\delta\sqrt{\sum_{i=1}^{k}\boldsymbol{y}_i^\top\boldsymbol{L}\boldsymbol{y}_i} = n^{1/2}\delta\|\boldsymbol{Y}\boldsymbol{B}^\top\|_F,
\end{aligned}$$

where the first inequality is obtained according to (13) and δ is given by (14). Applying the fact that $\mathrm{Tr}(\boldsymbol{L}^\dagger) = R(\mathcal{G})/n$ and $R(\mathcal{G}) \le n(n^2-1)/6$ [47], we have

$$\begin{aligned}
&n^{1/2}\delta\|\boldsymbol{Y}\boldsymbol{B}^\top\|_F \\
=&n^{1/2}\delta\Big(\sum_{(a,b)\in\mathcal{E}}\|\boldsymbol{Y}(\boldsymbol{e}_a-\boldsymbol{e}_b)\|^2\Big)^{1/2} \\
\le&n^{1/2}\delta\Big((1+\epsilon)\sum_{(a,b)\in\mathcal{E}}\|\boldsymbol{L}^\dagger(\boldsymbol{e}_a-\boldsymbol{e}_b)\|^2\Big)^{1/2} \\
=&n^{1/2}\delta\Big((1+\epsilon)\|\boldsymbol{L}^\dagger\boldsymbol{B}^\top\|_F^2\Big)^{1/2} \\
=&n^{1/2}\delta\left((1+\epsilon)\mathrm{Tr}(\boldsymbol{L}^\dagger\boldsymbol{B}^\top\boldsymbol{B}\boldsymbol{L}^\dagger)\right)^{1/2} \\
\le&n^{1/2}\delta\left((1+\epsilon)\mathrm{Tr}(\boldsymbol{L}^\dagger)\right)^{1/2} \\
\le&\delta\left(\frac{(1+\epsilon)n(n^2-1)}{6}\right)^{1/2}.
\end{aligned}$$

On the other hand, we provide a lower bound of $\|\boldsymbol{Y}\boldsymbol{e}_u\|^2$ as

$$\begin{aligned}
&\|\boldsymbol{Y}\boldsymbol{e}_u\|^2 \ge (1-\epsilon)\boldsymbol{e}_u^\top\boldsymbol{L}^{2\dagger}\boldsymbol{e}_u \\
=&(1-\epsilon)\left(\boldsymbol{e}_u-\frac{1}{n}\mathbf{1}\right)^\top\boldsymbol{L}^{2\dagger}\left(\boldsymbol{e}_u-\frac{1}{n}\mathbf{1}\right) \\
\ge&\frac{1-\epsilon}{\lambda_n^2}\frac{(n-1)^2}{n^2} \ge (1-\epsilon)\frac{(n-1)^2}{n^4}.
\end{aligned} \tag{18}$$

In (18), the first inequality is due to the following reason. Note that $\boldsymbol{e}_u - \frac{1}{n}\mathbf{1}$ is orthogonal to all-one vector $\mathbf{1}$, an eigenvector of $\boldsymbol{L}^\dagger$ associated with the only eigenvalue 0. Therefore, $(\boldsymbol{e}_u - \frac{1}{n}\mathbf{1})^\top\boldsymbol{L}^\dagger(\boldsymbol{e}_u-\frac{1}{n}\mathbf{1}) \ge \lambda_n^{-1}\left\|\boldsymbol{e}_u-\frac{1}{n}\mathbf{1}\right\|^2$ holds.

Combining the above-obtained results, it follows that

$$\begin{aligned}
&\frac{\left|\|\boldsymbol{Y}\boldsymbol{e}_u\| - \left\|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\right\|\right|}{\|\boldsymbol{Y}\boldsymbol{e}_u\|} \\
\le&\frac{\delta}{(1-\epsilon)^{1/2}}\frac{n^2}{n-1}\left(\frac{(1+\epsilon)n(n^2-1)}{6}\right)^{1/2} \le \frac{\epsilon}{3},
\end{aligned}$$

which is equivalent to (17). □

Having Lemma 3.1, we are in position to present our main theorem for approximating any diagonal entry of matrix $\boldsymbol{L}^{2\dagger}$.

Theorem 3.2: There is a $\widetilde{O}(m/\epsilon^2)$ expected time algorithm, which inputs an error parameter $0 < \epsilon \le 1/2$ and a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and returns a $(24\log n/\epsilon^2)\times n$ matrix $\tilde{\boldsymbol{Y}}$ such that with probability at least $1-1/n$,

$$(1-\epsilon)^2\boldsymbol{L}_{uu}^{2\dagger} \le \|\tilde{\boldsymbol{Y}}\boldsymbol{e}_u\|^2 \le (1+\epsilon)^2\boldsymbol{L}_{uu}^{2\dagger}$$

for any vertex $u \in \mathcal{V}$.

On the base of Theorem 3.2, an approximation algorithm Approx$\mathcal{H}$ is proposed for computing the second-order coherence $H_{\mathrm{SO}}(\mathcal{G})$ for an arbitrary graph $\mathcal{G}$ and $H_{\mathrm{SO}}(u)$ for all vertices $u \in \mathcal{V}$, which is presented in Algorithm 1. The input parameters of the algorithm are a connected undirected unweighted graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ and an approximation parameter ϵ. The output parameters are a set $\tilde{H}_{\mathrm{SO}}$ of an approximation of the second-order coherence $H_{\mathrm{SO}}(u)$ of each vertex u and an approximation of the second-order coherence $H_{\mathrm{SO}}(\mathcal{G})$ for graph $\mathcal{G}$. In Algorithm 1, $\sum_{i=1}^{k}\|\tilde{\boldsymbol{Y}}_{iu}\|^2$ is an approximation of the diagonal element $\boldsymbol{L}_{uu}^{2\dagger}$. Employing (11) and (12), we obtain guaranteed approximation for the second-order coherence of every node u and the whole graph $\mathcal{G}$.

IV. Numerical Experiments for Real and Model Networks

In this section, we present numerical results to evaluate the performance of the proposed approximation algorithm Approx$\mathcal{H}$. To this end, we perform extensive experiments for

ALGORITHM 1: Approx$\mathcal{H}(\mathcal{G},\epsilon)$

Input : $\mathcal{G}(\mathcal{V},\mathcal{E})$, ϵ
Output : $\tilde{H}_{\rm SO}=\{u,\tilde{H}_{\rm SO}(u)|u\in\mathcal{V}\}$, $\tilde{H}_{\rm SO}(\mathcal{G})$
$\boldsymbol{L}=$ Laplacian of graph $\mathcal{G}$,
Construct a matrix $\boldsymbol{Q}_{k\times n}$, where $k=\lceil 24\log n/\epsilon^2\rceil$ and each entry is $\pm 1/\sqrt{k}$ with identical probability
for $i=1$ *to* k **do**
  $\boldsymbol{q}_i^\top$=the i-th row of $\boldsymbol{Q}_{k\times n}$
  $\tilde{\boldsymbol{y}}_i=\texttt{LaplSolve}(\boldsymbol{L},\boldsymbol{q}_i,\delta)$ where parameter δ is given by (14)
end
for *each* $u\in\mathcal{V}$ **do**
  $\tilde{H}_{\rm SO}(u)=\frac{1}{2}\sum_{i=1}^{k}\|\tilde{\boldsymbol{Y}}_{iu}\|^2$
end
$\tilde{H}_{\rm SO}(\mathcal{G})=\frac{1}{n}\sum_{u\in\mathcal{V}}\tilde{H}_{\rm SO}(u)$
return $\tilde{H}_{\rm SO}=\{u,\tilde{H}_{\rm SO}(u)|u\in\mathcal{V}\}$ and $\tilde{H}_{\rm SO}(\mathcal{G})$

various real and model networks to demonstrate both the efficiency and accuracy of our algorithm Approx$\mathcal{H}$ for computing the second-order coherence $H_{\rm SO}(\mathcal{G})$ for an arbitrary graph $\mathcal{G}$ and $H_{\rm SO}(u)$ for all vertices in $\mathcal{G}$.

All our experiments are conducted on a Linux box with an *Intel i7-7700K @ 4.2-GHz (4 Cores)* and with *32GB* memory.The proposed approximation algorithm Approx$\mathcal{H}$ is implemented in *Julia v1.0.3*, which is a flexible dynamic language designed for high-performance scientific and numerical computing. Here, we use it to facilitate interactions with the Laplace Solver contained in the Laplace.jl package [49], the *Julia language* source code for which is publicly available [1].

A. Results for Coherence of Real Networks

In this subsection, we evaluate the performance of our approximation algorithm Approx$\mathcal{H}$ on different realistic networks representably chosen from various domains, which are from KONECT — The Koblenz Network Collection [50]. Since the coherence is defined only for connected graphs, for those real networks with multiple components, we perform experiments on the largest connected component (LCC) for each of these networks. Table I reports related information of considered real networks. For a network with n vertices and m edges, we denote the number of vertices and edges in its largest connected component by n' and m', respectively. In Table I, networks are listed in an increasing order of the number of vertices in original networks. As displayed in Table I, the vertex number for studied real-world networks ranges from about 200 to more than 1.3×10^6, which is enough for the purpose of performance evaluation.

We first consider the efficiency of our approximation algorithm Approx$\mathcal{H}$ for computing the average coherence $H_{\rm SO}(\mathcal{G})$ of real networks. For this purpose, we compare it with a direct accurate algorithm called Exact$\mathcal{H}$, which computes $H_{\rm SO}(\mathcal{G})$ via (12) by inverting $\boldsymbol{L}^2+\frac{1}{n}\boldsymbol{J}$. To objectively evaluate the efficiency of algorithms Approx$\mathcal{H}$ and Exact$\mathcal{H}$, both algorithms are run on a single thread, and their CPU time is reported in Table II. From Table II, we observe that for moderate approximation parameter ϵ, the CPU time for Approx$\mathcal{H}$ is much less than that for Exact$\mathcal{H}$, especially for large networks tested. Note that the direct algorithm Exact$\mathcal{H}$ requires much more memory and time, it is not applicable to the last seven networks marked with "$*$" in Tables I and II, for which the vertex number ranges from 10^5 to 10^6. In comparison with Exact$\mathcal{H}$, Approx$\mathcal{H}$ we can compute the approximate values for the coherence of those networks within at most several hours even for $\epsilon=0.05$. Therefore, compared with exact algorithm Exact$\mathcal{H}$, our approximated algorithm Approx$\mathcal{H}$ significantly improves the efficiency and is scalable to large networks.

We proceed to show that in addition to the high efficiency, our approximation algorithm Approx$\mathcal{H}$ also provides a good approximation $\tilde{H}_{\rm SO}(u)$ for the vertex coherence $H_{\rm SO}(u)$. To show the accuracy for Approx$\mathcal{H}$, we compare the approximate values $\tilde{H}_{\rm SO}(u)$ computed by Approx$\mathcal{H}$ with the exact results of $H_{\rm SO}(u)$ computed by Exact$\mathcal{H}$, and present the mean relative errors σ in Table III, which is defined by $\sigma=\frac{1}{n}\sum_{u\in\mathcal{V}}|H_{\rm SO}(u)-\tilde{H}_{\rm SO}(u)|/H_{\rm SO}(u)$. Table III indicates that the actual mean relative errors σ for the studied networks and all $\epsilon=0.3,0.2,0.1,0.05$ are less than 10% and almost negligible (1% or so) for $\epsilon=0.05$. Moreover, for all networks we tested, σ are magnitudes smaller than the theoretical guarantee provided by Theorem 3.2. Consequently, our algorithm Approx$\mathcal{H}$ provides a very desirable approximation for network coherence in practice.

B. Results for Network Coherence of Model Networks

For an arbitrary graph $\mathcal{G}$, exact expression for the second-order network coherence $H_{\rm SO}(\mathcal{G})$ is quite difficult and even impossible. However, for some deterministically growing model networks, we can obtain explicit formulas for this interesting quantity. For example, the second-order network coherence has been exactly determined for the hierarchical graphs [51], [15], Sierpiński graphs [15], and Koch networks [25]. In this subsection, we utilize these obtained exact results to further evaluate the efficiency and accuracy of our

TABLE I
STATISTICS OF THE COLLECTION OF DATASETS USED IN OUR EXPERIMENTS.

Network	n	m	n'	m'
Jazz musicians	198	2,742	198	2,742
Chicago	1,467	1,298	823	822
Hamster full	2,426	16,631	2,000	16,098
Facebook (NIPS)	4,039	88,234	4,039	88,234
CA-GrQc	5,242	14,496	4,158	13,422
Reactome	6,327	147,547	5,973	145,778
Route views	6,474	13,895	6,474	12,572
Pretty Good Privacy	10,680	24,316	10,680	24,316
CA-HepPh	12,008	118,521	11,204	117,619
Astro-ph	18,772	198,110	17,903	196,972
CAIDA	26,475	53,381	26,475	53,381
Brightkite	58,228	214,078	56,739	212,945
Livemocha*	104,103	2,193,083	104,103	2,193,083
WordNet*	146,005	656,999	145,145	656,230
Gowalla*	196,591	950,327	196,591	950,327
com-DBLP*	317,080	1,049,866	317,080	1,049,866
Amazon*	334,863	925,872	334,863	925,872
Pennsylvania*	1,088,092	1,541,898	1,087,562	1,541,514
roadNet-TX*	1,379,917	1,921,660	1,351,137	1,879,201

[1] http://danspielman.github.io/Laplacians.jl/latest/

TABLE II
THE RUNNING TIME (SECONDS, s) OF EXACT$\mathcal{H}$ AND APPROX$\mathcal{H}$ WITH VARIOUS ϵ ON VARIOUS REALISTIC NETWORKS.

Network	Exact$\mathcal{H}$	Approx$\mathcal{H}$ (s) with various ϵ					
	(s)	0.3	0.25	0.2	0.15	0.1	0.05
Jazz musicians	0.001	0.016	0.022	0.030	0.048	0.103	0.397
Chicago	0.034	0.007	0.009	0.014	0.023	0.051	0.197
Hamster full	0.369	0.171	0.228	0.318	0.550	1.392	5.462
Facebook (NIPS)	3.042	0.721	1.047	1.625	2.462	5.652	23.44
CA-GrQc	3.017	0.241	0.337	0.520	0.961	2.086	8.578
Reactome	9.047	1.235	1.900	2.596	4.409	10.42	42.45
Route views	11.26	0.219	0.278	0.457	0.877	1.876	7.176
Pretty Good Privacy	47.17	0.681	0.797	1.718	2.252	5.282	22.63
CA-HepPh	55.60	1.396	1.916	3.171	5.089	11.01	46.16
Astro-ph	228.4	3.241	4.266	7.088	11.43	25.46	97.69
CAIDA	739.8	1.271	1.534	2.275	4.428	9.557	35.92
Brightkite	7274	8.538	7.055	11.75	19.96	44.47	175.0
Livemocha*	–	44.25	62.55	94.44	159.1	384.0	1442
WordNet*	–	19.96	27.02	44.25	79.06	169.0	742.6
Gowalla*	–	33.76	43.82	67.49	118.5	272.3	1139
com-DBLP*	–	61.75	87.98	132.7	237.2	516.0	2136
Amazon*	–	88.50	124.4	205.7	352.4	760.1	2925
Pennsylvania*	–	353.5	496.5	764.2	1323	3057	12260
roadNet-TX*	–	482.7	729.3	1125	1955	4461	17601

TABLE III
MEAN RELATIVE ERROR σ ($\times 10^{-2}$) OF $\tilde{H}_{\rm SO}(u)$ FOR ALL VERTICES u IN DIFFERENT REAL NETWORKS.

Network	Mean relative error for various ϵ			
	0.3	0.2	0.1	0.05
Jazz musicians	9.16	8.01	3.16	1.44
Chicago	7.56	4.77	2.89	1.71
Hamster full	7.09	4.33	2.25	1.15
Facebook (NIPS)	7.97	8.20	3.84	0.84
CA-GrQc	8.50	5.24	2.60	1.27
Reactome	5.67	5.93	2.30	1.11
Route views	4.35	2.91	1.45	0.74
Pretty Good Privacy	7.92	5.69	2.69	1.30
CA-HepPh	6.01	4.14	2.08	1.12
Astro-ph	5.77	3.82	1.84	0.93
CAIDA	3.69	2.47	1.23	0.62
Brightkite	4.56	2.99	1.55	0.76

proposed algorithm Approx$\mathcal{H}$ approximating the second-order network coherence.

1) Hierarchical Graphs and Their Network Coherence: The hierarchical graphs are constructed based on the hierarchical product of graphs introduced first by Godsil and McKay [52].

Definition 4.1: [52] Let $\mathcal{G}_1(\mathcal{V}_1, \mathcal{E}_1)$ and $\mathcal{G}_2(\mathcal{V}_2, \mathcal{E}_2)$ be two graphs, with a root vertex labeled by 1. Then, the hierarchical product $\mathcal{G}_2 \sqcap \mathcal{G}_1$ of $\mathcal{G}_1$ and $\mathcal{G}_2$ is a graph with vertices x_2x_1, $x_i \in \mathcal{V}_i$ $(i = 1, 2)$, and edges (x_2x_1, y_2y_1), where either $y_2 = x_2$ and y_1 and x_1 are adjacent in $\mathcal{G}_1$ or $y_1 = x_1 = 1$ and y_2 and x_2 are adjacent in $\mathcal{G}_2$.

Using the associative property of hierarchical product, we can create the hierarchical graphs by iteratively applying the operation of hierarchical product to the complete graphs [53].

Definition 4.2: [53] Let $\mathcal{Q}_k$ be the complete graph with k ($k \geq 3$) vertices, labelled by 1, 2, ..., k. And let $\mathcal{H}_{g,k}$ be the hierarchical graphs $\mathcal{H}_{g,k}$, $g \geq 1$, at generation g. Initially, $\mathcal{H}_{1,k}$ is isomorphic to $\mathcal{Q}_k$. For $g > 1$, $\mathcal{H}_{g,k}$ is the hierarchical product of g copies of $\mathcal{Q}_k$, namely, $\mathcal{H}_{g,k} = \mathcal{H}_{g-1,k} \sqcap \mathcal{Q}_k = \mathcal{Q}_k \sqcap \overset{g}{\cdots} \sqcap \mathcal{Q}_k$.

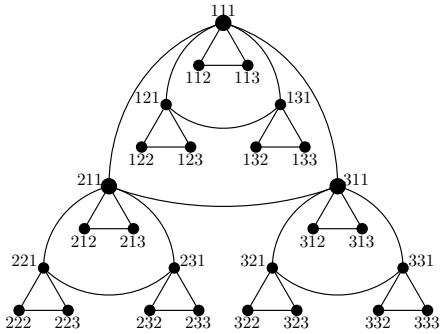


Fig. 1. The hierarchical graph $\mathcal{H}_{3,3}$ and its vertex labelling.

Fig. 1 illustrates the hierarchical graph $\mathcal{H}_{3,3}$ and its vertex labelling.

In the hierarchical graph $\mathcal{H}_{g,k}$, there are k^g vertices and $\frac{k^{g+1}-k}{2}$ edges. It was shown [15] that the network coherence for $\mathcal{H}_{g,k}$ is

$$H_{\rm SO}(\mathcal{H}_{g,k}) = \frac{k - k^2 + k^g(k^2 - 5k - 6) + k^{2g}(4k + 6)}{2k^{g+3}(1 + k)} + \frac{2g(1 - k)}{k^3}. \quad (19)$$

2) Sierpiński Graphs and Their Network Coherence: The Sierpiński graphs introduced by Klavžar and Milutinović [54] are also constructed in an iterative way, which are a generalization of the Tower of Hanoi graph [55], [56]. Let $\mathcal{S}_{g,k}$ be the Sierpiński graphs $\mathcal{S}_{g,k}$, $g \geq 1$, at iteration g. The vertex set, denoted by $\mathcal{V}(\mathcal{S}_{g,k})$, of $\mathcal{S}_{g,k}$ consists of all g-tuples of integers $1, 2, \cdots, k$, that is, $\mathcal{V}(\mathcal{S}_{g,k}) = \{1, 2, \cdots, k\}^g$. All vertices in $\mathcal{S}_{g,k}$ can be labelled in the form $u_1u_2 \cdots u_g$, where $u_i \in \{1, 2, \cdots, k\}$ for all $i = 1, 2, \cdots, g$. Two vertices $p = p_1p_2 \cdots p_n$ and $q = q_1q_2 \cdots q_n$ are directly connected by an edge if and only if there exists a positive integer h $(1 \leq h \leq g)$ such that

(a) $p_i = q_i$ for $1 \leq i \leq h - 1$;
(b) $p_h \neq q_h$;
(c) $p_i = q_h$ and $q_i = p_h$ for $h + 1 \leq i \leq g$.

Fig. 2 illustrates the Sierpiński graph $\mathcal{S}_{3,3}$ and its vertex labelling.

The vertex number and the edge number for the Sierpiński graphs $\mathcal{S}_{g,k}$ are identical to those corresponding to the hierarchical graphs $\mathcal{H}_{g,k}$. And it has been shown [15] that the network coherence for $\mathcal{S}_{g,k}$ is

$$H_{\rm SO}(\mathcal{S}_{g,k}) = -\frac{7k^2 + 13k + 2}{2k^{g+1}(k+1)^2(k+2)^2(k+3)} - \frac{(k-2)(k^3 + 4k^2 + 4k + 2)}{2k^2(k+2)^2(k^2 + 3k + 4)} + \frac{(k-1)(k+2)^{g-2}(k^2 + k + 2)}{2k^{g+1}(k+1)^2} + \frac{k^5 + 7k^4 + 16k^3 + 28k^2 + 26k + 12}{2k^{g+2}(k+1)^2(k+3)(k^2 + 3k + 4)} \times (k-1)(k+2)^{2g-2}. \quad (20)$$

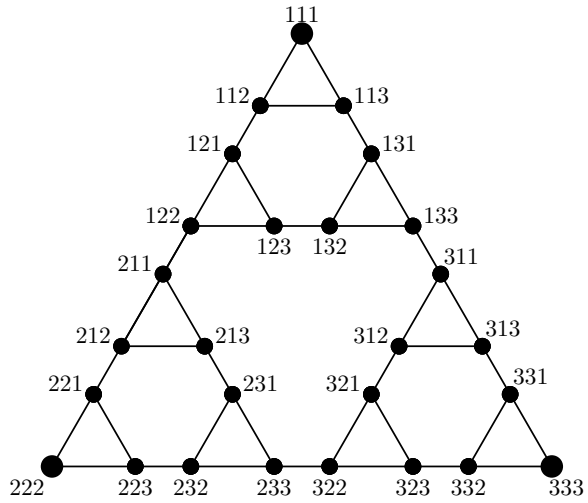


Fig. 2. The Sierpiński graph $\mathcal{S}_{3,3}$ and its vertex labeling.

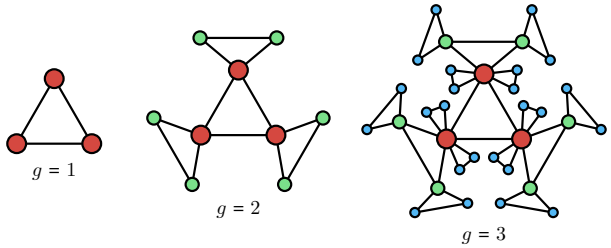


Fig. 3. Construction process for the Koch networks.

3) Koch Networks and Their Network Coherence: The Koch networks are also built iteratively [57]. Let $\mathcal{K}_g$ ($g \geq 1$) be the Koch network at iteration g. For $g = 1$, $\mathcal{K}_1$ is a triangle consisting of three vertices and three edges. For $g \geq 2$, $\mathcal{K}_g$ is derived from $\mathcal{K}_{g-1}$ in the following way. For each of the three vertices in every existing triangle in $\mathcal{K}_{g-1}$, two new vertices are created, both of which and their "mother" vertices are linked to one another forming a new triangle. Fig. 3 illustrates the first several iterative steps of the Koch networks. It is easy to check that in network $\mathcal{K}_g$, the number of vertices is $2 \cdot 4^{g-1} + 1$, and the number of edges is $3 \cdot 4^{g-1}$.

It has be explicitly determined [25] that the second-order network coherence for the Koch network $\mathcal{K}_g$ is

$$H_{\mathrm{SO}}(\mathcal{K}_g) = \frac{139 \cdot 2^{8g-7} + 2^{6g-4}(45g^2 - 15g + 1)}{405 \cdot (2^{2g-1} + 1)^3} + \frac{2^{4g-4}(180g^2 - 120g - 53) - 2^{2g}}{405 \cdot (2^{2g-1} + 1)^3}. \tag{21}$$

4) Approximate Results for Network Coherence of Deterministic Networks: To further display the performance of our algorithm Approx$\mathcal{H}$ for approximating network coherence, we apply it to calculate the second-order network coherence for the above three deterministic networks: hierarchical graphs, Sierpiński graphs, and Koch networks. In Table IV, we report the exact values for network coherence $H_{\mathrm{SO}}(\mathcal{G})$, our numerical approximation results $\tilde{H}_{\mathrm{SO}}(\mathcal{G})$ and their relative error $\rho = (H_{\mathrm{SO}}(\mathcal{G}) - \tilde{H}_{\mathrm{SO}}(\mathcal{G}))/H_{\mathrm{SO}}(\mathcal{G})$, and running time (seconds, s) of our algorithm Approx$\mathcal{H}$. The exact second-order coherence $H_{\mathrm{SO}}(\mathcal{G})$ is obtained via (19), (20) and (21), while their approximation $\tilde{H}_{\mathrm{SO}}(\mathcal{G})$ is obtained through algorithm Approx$\mathcal{H}$ with $\epsilon = 0.05$. Table IV indicates that algorithm Approx$\mathcal{H}$ works effectively and efficiently for all the three considered networks. This once again demonstrates the advantage of our proposed algorithm Approx$\mathcal{H}$ for approximating network coherence of large networks with millions of vertices.

TABLE IV
MODEL NETWORKS AND THEIR EXACT AND APPROXIMATE COHERENCE.

Network	Vertices	Edges	H_{SO}	$\tilde{H}_{\mathrm{SO}}$	ρ	Time
$\mathcal{H}_{13,3}$	1,594,323	2,391,483	132,858	133,907	0.0078	2113
$\mathcal{S}_{13,3}$	1,594,323	2,391,483	3,115,462,641	3,156,676,337	0.013	36616
$\mathcal{K}_{11}$	2,097,153	3,145,728	89,977	90,183	0.0023	2704

C. Experiments for Vertex Coherence of Paths and Cycles

In the proceeding subsections, we show the performance of our algorithm Approx$\mathcal{H}$ for approximating second-order network coherence. In this subsection, we display the efficiency and accuracy of Approx$\mathcal{H}$ for calculating the vertex coherence for the path and cycle graphs, for both of which the vertex coherence for each vertex can be obtained explicitly [28].

The path graph $\mathcal{P}_n$ is a graph with n vertices labeled $\{1, 2, \cdots, n\}$, where vertex 1 is at one end of the path and the remaining vertices are labeled by their graph distances to vertex 1. In $\mathcal{P}_n$, the exact value of second-order vertex coherence for vertex $j \in \{1, 2, \cdots, n\}$ is [28]

$$H_{\mathrm{SO}}(j) = \frac{1}{360n}(4n^4 - (60j^2 - 60j + 5)n^2 + 60j(j-1)(2j-1)n - 60j^2(j-1)^2 + 1). \tag{22}$$

The n-vertex cycle graph $\mathcal{C}_n$ is obtained from $\mathcal{P}_n$ by adding an edge connecting the two end vertices 1 and n in $\mathcal{P}_n$. By definition, the structural and dynamical properties of all vertices in $\mathcal{C}_n$ are equivalent to one another. It was shown [28] that for any vertex $j \in \mathcal{C}_n$, the vertex coherence is the same and reads

$$H_{\mathrm{SO}}(j) = \frac{1}{1440}\left(n^3 + 10n - \frac{11}{n}\right). \tag{23}$$

We also use our approximation algorithm Approx$\mathcal{H}$ to compute the vertex coherence of vertex 1 for the path and cycle with three million vertices. The experimental results are displayed in Table V, where we report the exact coherence $H_{\mathrm{SO}}(1)$, its approximation $\tilde{H}_{\mathrm{SO}}(1)$, the relative error $\rho = |H_{\mathrm{SO}}(1) - \tilde{H}_{\mathrm{SO}}(1)|/H_{\mathrm{SO}}(1)$ and running time (seconds, s) for the path $\mathcal{P}_{3,000,000}$ and the cycle $\mathcal{C}_{3,000,000}$. $H_{\mathrm{SO}}(1)$ is obtained via (22) and (23), while $\tilde{H}_{\mathrm{SO}}(1)$ is obtained through algorithm Approx$\mathcal{H}$ with $\epsilon = 0.05$. As shown in Table V, for both networks, the running time is less than two hours, and the relative error ρ for $\tilde{H}_{\mathrm{SO}}(1)$ is strictly below the theoretical bound. Thus, our algorithm Approx$\mathcal{H}$ is also efficient and accurate for approximating vertex coherence of a single vertex.

TABLE V
EXACT AND APPROXIMATE VERTEX COHERENCE OF VERTEX 1 FOR THE PATH AND CYCLE.

Network	H_{SO}	$\tilde{H}_{\mathrm{SO}}$	ρ	Time
$\mathcal{P}_{3,000,000}$	3.0000×10^{17}	3.0259×10^{17}	0.0086	5730
$\mathcal{C}_{3,000,000}$	1.8750×10^{16}	1.8704×10^{16}	0.0025	6305

V. CONCLUSIONS

It has been established [28] that for second-order consensus algorithm in a noisy network, its robustness can be measured by vertex coherence and network coherence, with the former being the steady-state variance of the deviation of each vertex state from the average and the latter being the average steady-state variance of the system. Both performance measures depend on the biharmonic distances in the communication graph. However, for a general graph, a naïve method for computing biharmonic distances requires inverting a matrix that has cubic time complexity and is infeasible for large networks. In this paper, by using the implicit connection between the robustness measures and the diagonals of pseudoinverse $\boldsymbol{L}^{2\dagger}$ of the square for Laplacian matrix associated with the communication graph, we proposed a computationally inexpensive and accurate algorithm that approximates all diagonal entries of $\boldsymbol{L}^{2\dagger}$ in nearly linear time with regard to the edge number of the graph. Extensive numerical experiments on various realistic and model networks demonstrate the efficiency and accuracy of our proposed approximation algorithm.

It is worth pointing out that although we only considered binary networks, our approximation approach is not difficult to be extended to weighted communication graphs [58], [59] by considering the weight of every edge as its conductance. In addition, the technique and process of our algorithm can be easily modified to address second-order noisy consensus dynamics, where some nodes are chosen to be leaders and are noise-corrupted [60]. Finally, our algorithm can be modified and carried out in a distributed way in order to further reduce the time and space costs, although it has obtained desired effect even on a single machine. For example, in Algorithm 1 $\tilde{\boldsymbol{Y}}^{\top} = [\tilde{\boldsymbol{y}}_1, \tilde{\boldsymbol{y}}_2, ..., \tilde{\boldsymbol{y}}_k]$ are computed in the first for loop, which actually can be done in parallel, leading to improvement of running time. ¡¡¡¡

REFERENCES

[1] D. Mateo, N. Horsevad, V. Hassani, M. Chamanbaz, and R. Bouffanais, "Optimal network topology for responsive collective behavior," *Sci. Adv.*, vol. 5, no. 4, p. eaau0999, 2019.

[2] J. H. Fowler and N. A. Christakis, "Cooperative behavior cascades in human social networks," *Proc. Natl. Acad. Sci.*, vol. 107, no. 12, pp. 5334–5338, 2010.

[3] T. Vicsek and A. Zafeiris, "Collective motion," *Phys. Rep.*, vol. 517, no. 3-4, pp. 71–140, 2012.

[4] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[5] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Trans. Autom. Control*, vol. 51, no. 3, pp. 401–420, Mar. 2006.

[6] D. V. Dimarogonas and K. J. Kyriakopoulos, "On the rendezvous problem for multiple nonholonomic agents," *IEEE Trans. Autom. Control*, vol. 52, no. 5, pp. 916–922, May 2007.

[7] Q. Li and D. Rus, "Global clock synchronization in sensor networks," *IEEE Trans. Comput.*, vol. 55, no. 2, pp. 214–226, 2006.

[8] W. Yu, G. Chen, Z. Wang, and W. Yang, "Distributed consensus filtering in sensor networks," *IEEE Trans. Syst., Man, Cybern., B Cybern.*, vol. 39, no. 6, pp. 1568–1577, 2009.

[9] S. Zhu, C. Chen, W. Li, B. Yang, and X. Guan, "Distributed optimal consensus filter for target tracking in heterogeneous sensor networks," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1963–1976, 2013.

[10] S. Motsch and E. Tadmor, "Heterophilious dynamics enhances consensus," *SIAM Rev.*, vol. 56, no. 4, pp. 577–621, 2014.

[11] X. Wu, Y. Tang, J. Cao, and W. Zhang, "Distributed consensus of stochastic delayed multi-agent systems under asynchronous switching," *IEEE Trans. Cybern.*, vol. 46, no. 8, pp. 1817–1827, 2016.

[12] X. Shi, J. Cao, and W. Huang, "Distributed parametric consensus optimization with an application to model predictive consensus problem," *IEEE Trans. Cybern.*, vol. 48, no. 7, pp. 2024–2035, 2017.

[13] A. Olshevsky and J. N. Tsitsiklis, "Convergence speed in distributed consensus and averaging," *SIAM J. Control Optim.*, vol. 48, no. 1, pp. 33–55, 2009.

[14] T. C. Aysal and K. E. Barner, "Convergence of consensus models with stochastic disturbances," *IEEE Trans. Inf. Theory*, vol. 56, no. 8, pp. 4101–4113, 2010.

[15] Y. Qi, Z. Zhang, Y. Yi, and H. Li, "Consensus in self-similar hierarchical graphs and Sierpiński graphs: Convergence speed, delay robustness, and coherence," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 592–603, 2019.

[16] F. Xiao and L. Wang, "Consensus protocols for discrete-time multi-agent systems with time-varying delays," *Automatica*, vol. 44, no. 10, pp. 2577–2582, 2008.

[17] U. Münz, A. Papachristodoulou, and F. Allgöwer, "Delay robustness in consensus problems," *Automatica*, vol. 46, no. 8, pp. 1252–1265, 2010.

[18] L. Xiao, S. Boyd, and S.-J. Kim, "Distributed average consensus with least-mean-square deviation," *J. Parallel .Dist. Comput.*, vol. 67, no. 1, pp. 33–46, Jan. 2007.

[19] B. Bamieh, M. Jovanovic R, P. Mitra, and S. Patterson, "Effect of topological dimension on rigidity of vehicle formations: Fundamental limitations of local feedback," in *Proc. 47th IEEE Conf. Decision Control*, Dec. 2008, pp. 369–374.

[20] G. F. Young, L. Scardovi, and N. E. Leonard, "Robustness of noisy consensus dynamics with directed communication," in *Proc. Amer. Control Conf.*, Jun. 2010, pp. 6312–6317.

[21] S. Patterson and B. Bamieh, "Leader selection for optimal network coherence," in *Proc. 49th IEEE Conf. Decision Control*. IEEE, 2010, pp. 2692–2697.

[22] B. Bamieh, M. R. Jovanovic, P. Mitra, and S. Patterson, "Coherence in large-scale networks: Dimension-dependent limitations of local feedback," *IEEE Trans. Autom. Control*, vol. 57, no. 9, pp. 2235–2249, Sep. 2012.

[23] S. Patterson and B. Bamieh, "Consensus and coherence in fractal networks," *IEEE Trans. Control Netw. Syst.*, vol. 1, no. 4, pp. 338–348, Sep. 2014.

[24] Y. Yi, Z. Zhang, Y. Lin, and G. Chen, "Small-world topology can significantly improve the performance of noisy consensus in a complex network," *Comput. J.*, vol. 58, no. 12, pp. 3242–3254, 2015.

[25] Y. Yi, Z. Zhang, L. Shan, and G. Chen, "Robustness of first-and second-order consensus algorithms for a noisy scale-free small-world Koch network," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 1, pp. 342–350, 2017.

[26] Y. Yi, Z. Zhang, and S. Patterson, "Scale-free loopy structure is resistant to noise in consensus dynamics in complex networks," *IEEE Trans. Cybern.*, vol. 50, no. 1, pp. 190–200, 2020.

[27] D. J. Klein and M. Randić, "Resistance distance," *J. Math. Chem.*, vol. 12, no. 1, pp. 81–95, 1993.

[28] Y. Yi, B. Yang, Z. Zhang, and S. Patterson, "Biharmonic distance and performance of second-order consensus networks with stochastic disturbances," in *Proc. Amer. Control Conf.* IEEE, 2018, pp. 4943–4950.

[29] Y. Lipman, R. M. Rustamov, and T. A. Funkhouser, "Biharmonic distance," *ACM Trans. Graph.*, vol. 29, no. 3, p. 27, 2010.

[30] W. B. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," *Contemp. Math.*, vol. 26, pp. 189–206, 1984.

[31] D. Achlioptas, "Database-friendly random projections," in *Proc. 20th ACM Symp. Principles of Database Syst.* ACM, 2001, pp. 274–281.

[32] ——, "Database-friendly random projections: Johnson-Lindenstrauss with binary coins," *J. Comput. Syst. Sci.*, vol. 66, no. 4, pp. 671–687, 2003.

[33] D. A. Spielman and S.-H. Teng, "Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems," in *Proc. 36th Ann. ACM Symp. Theory Comput.* ACM, 2004, pp. 81–90.

[34] ——, "Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems," *SIAM J. Matrix Anal. Appl.*, vol. 35, no. 3, pp. 835–885, 2014.

[35] I. Koutis, G. L. Miller, and R. Peng, "A nearly-m log n time solver for SDD linear systems," in *Proc. IEEE 52nd Ann. Symp. Found. Comput. Sci.* IEEE, 2011, pp. 590–598.

[36] O. E. Livne and A. Brandt, "Lean algebraic multigrid (LAMG): Fast graph Laplacian linear solver," *SIAM J. Sci. Comput.*, vol. 34, no. 4, pp. B499–B522, 2012.

[37] M. B. Cohen, R. Kyng, G. L. Miller, J. W. Pachocki, R. Peng, A. B. Rao, and S. C. Xu, "Solving SDD linear systems in nearly m log 1/2 n time," in *Proc. 46th Ann. ACM Symp. Theory Comput.* ACM, 2014, pp. 343–352.

[38] R. Grone and R. Merris, "The Laplacian spectrum of a graph II," *SIAM J. Discrete Math.*, vol. 7, no. 2, pp. 221–229, 1994.

[39] A. Ben-Israel and T. N. E. Greville, *Generalized inverses: theory and applications*. J. Wiley, 1974.

[40] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 355–369, 2007.

[41] A. Ghosh, S. Boyd, and A. Saberi, "Minimizing effective resistance of a graph," *SIAM Rev.*, vol. 50, no. 1, pp. 37–66, Feb. 2008.

[42] W. Xiao and I. Gutman, "Resistance distance and Laplacian spectrum," *Theoret. Chem. Acc.*, vol. 110, no. 4, pp. 284–289, 2003.

[43] U. Brandes and D. Fleischer, "Centrality measures based on current flow," in *Proc. Ann. Symp. Theoret. Aspects Comput. Sci.*, vol. 3404. Springer, 2005, pp. 533–544.

[44] M. Tyloo, T. Coletta, and P. Jacquod, "Robustness of synchrony in complex networks and generalized Kirchhoff indices," *Phys. Rev. Lett.*, vol. 120, no. 8, p. 084101, 2018.

[45] P. G. Doyle and J. L. Snell, *Random Walks and Electric Networks*. Mathematical Association of America, 1984.

[46] E. Bozzo and M. Franceschet, "Resistance distance, closeness, and betweenness," *Social Networks*, vol. 35, no. 3, pp. 460–469, 2013.

[47] W. S. Lovejoy and C. H. Loch, "Minimal and maximal characteristic path lengths in connected sociomatrices," *Social Networks*, vol. 25, no. 4, pp. 333–347, 2003.

[48] K. Fitch and N. E. Leonard, "Joint centrality distinguishes optimal leaders in noisy networks," *IEEE Trans. Control Netw. Syst.*, vol. 3, no. 4, pp. 366–378, 2016.

[49] R. Kyng and S. Sachdeva, "Approximate Gaussian elimination for Laplacians-fast, sparse, and simple," in *Proc. IEEE 57th Ann. Symp. Found. Comput. Sci.* IEEE, 2016, pp. 573–582.

[50] J. Kunegis, "Konect: The koblenz network collection," in *Proc. 22nd Int. Conf. World Wide Web*. New York, USA: ACM, 2013, pp. 1343–1350.

[51] Y. Qi, Y. Yi, and Z. Zhang, "Topological and spectral properties of small-world hierarchical graphs," *Comput. J.*, vol. 62, no. 5, pp. 769–784, 2019.

[52] C.-D. Godsil and B.-D. McKay, "A new graph product and its spectrum," *Bull. Austral. Math. Soc.*, vol. 18, no. 01, pp. 21–28, 1978.

[53] L. Barrière, F. Comellas, C. Dalfó, and M. A. Fiol, "The hierarchical product of graphs," *Discrete Appl. Math.*, vol. 157, pp. 36–48, 2009.

[54] S. Klavžar and U. Milutinović, "Graphs $S(n, k)$ and a variant of the Tower of Hanoi problem," *Czech. Math. J.*, vol. 47, no. 1, pp. 95–104, 1997.

[55] Z. Zhang, S. Wu, M. Li, and F. Comellas, "The number and degree distribution of spanning trees in the Tower of Hanoi graph," *Theoret. Comput. Sci.*, vol. 609, pp. 443–455, 2016.

[56] Y. Jin, H. Li, and Z. Zhang, "Maximum matchings and minimum dominating sets in Apollonian networks and extended Tower of Hanoi graphs," *Theoret. Comput. Sci.*, vol. 703, pp. 37–54, 2017.

[57] Z. Zhang, S. Zhou, W. Xie, L. Chen, Y. Lin, and J. Guan, "Standard random walks and trapping on the Koch network with scale-free behavior and small-world effect," *Phys. Rev. E*, vol. 79, no. 6, p. 061113, 2009.

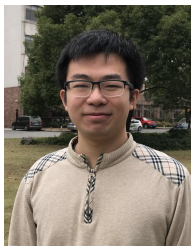
[58] A. Barrat, M. Barthelemy, R. Pastor-Satorras, and A. Vespignani, "The architecture of complex weighted networks," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 101, no. 11, pp. 3747–3752, 2004.

[59] Y. Qi, H. Li, and Z. Zhang, "Extended corona product as an exactly tractable model for weighted heterogeneous networks," *Comput. J.*, vol. 61, no. 5, pp. 745–760, 2018.

[60] E. Mackin and S. Patterson, "Second order consensus with absolute information," in *Proc. IEEE Conf. Decision and Control*. IEEE, 2018, pp. 4523–4528.

Zuobai Zhang is currently an undergraduate student working toward the B.S. degree in School of Computer Science, Fudan University, Shanghai, China. His research interests include graph algorithms, social networks, and network science.

Wanyue Xu (S'20) received the B.Eng. degree in computer science and technology, Shangdong University, Weihai, China, in 2019. She is currently pursuing the Master's degree in School of Computer Science, Fudan University, Shanghai, China. Her research interests include network science, graph data mining, social network analysis, and random walks. She is a student member of the IEEE.



Yuhao Yi received the B.S. degree and the Ph.D. degree in computer science from Fudan University, Shanghai, China, in 2013 and 2019, respectively. From 2016 to 2017, he was a Visiting Scholar with the Rensselaer Polytechnic Institute, Troy, NY, USA, where he is currently a Post-Doctoral Research Fellow. His research interests include network science and distributed control. His current research interests includes analyzing and optimizing the performance of averaging consensus dynamics in large-scale networks.

Zhongzhi Zhang (M'19) received the B.Sc. degree in applied mathematics from Anhui University, Hefei, China, in 1997 and the Ph.D. degree in management science and engineering from the Dalian University of Technology, Dalian, China, in 2006. From 2006 to 2008, he was a Post-Doctoral Research Fellow with Fudan University, Shanghai, China, where he is currently a Full Professor with the School of Computer Science. He has published over 140 papers in international journals or conferences in the field of network modeling and dynamics. He has over 2700 ISI Web of Science citations with an H-index of 31 according to the Clarivate. His current research interests include network science, graph data mining, social network analysis, spectral graph theory, and random walks. Dr. Zhang was a recipient of the Excellent Doctoral Dissertation Award of Liaoning Province, China, in 2007, the Excellent Post-Doctor Award of Fudan University in 2008, and the Shanghai Natural Science Award (third class) in 2013. He is a member of the IEEE.