Information Sciences 679 (2024) 121096

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Generic network sparsification via degree- and subgraph-based edge sampling ☆

Zhen Su [a,b,*], Yang Liu [c], Jürgen Kurths [a,d], Henning Meyerhenke [b,**]

[a] Potsdam Institute for Climate Impact Research, Potsdam, Germany
[b] Department of Computer Science, Humboldt-Universität zu Berlin, Berlin, Germany
[c] School of Artificial Intelligence, Optics and Electronics (iOPEN), Northwestern Polytechnical University, Shaanxi, China
[d] Department of Physics, Humboldt-Universität zu Berlin, Berlin, Germany

ARTICLE INFO

Keywords:
Graph sparsification
Edge sampling
Network analysis
Triads

ABSTRACT

Network (or graph) sparsification accelerates many downstream analyses. For graph sparsification, sampling methods derived from local heuristic considerations are common in practice, due to their efficiency in generating sparse subgraphs using only local information. Filtering-based edge sampling is the most typical approach in this respect, yet it heavily depends on an appropriate definition of edge importance. Instead, we propose a generalized node-focused edge sampling framework by preserving scaled/expected local *node* characteristics. Apart from expected degrees, these local node characteristics include the expected number of triangles and the expected number of non-closed wedges associated with a node. From a technical point of view, we adapt a game-theoretic sampling method from uncertain graph generation to obtain sparse subgraphs that approximate the expected local properties. We include a tolerance threshold for much faster convergence. Within this framework, we provide appropriate algorithmic variants for sparsification. Moreover, we propose a network measure called *tri-wedge assortativity* for the selection of the most suitable variant when sparsifying a given network. Extensive experimental studies on functional climate, observed real-world, and synthetic networks show the effectiveness of our method in preserving overall structural network properties – on average consistently better than the state of the art.

1. Introduction

Networks (= graphs, we use both terms interchangeably) have become generic data representations in various domains, ranging from sociology to biology and even climatology [2]. Networks can be categorized based on the underlying process they model. Real-world networks can model relationships among physical objects (e.g., in infrastructure such as streets, gas, and water) or conceptual

☆ A preliminary version of this article was published in the Proceedings of the 2022 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) [1].

* Corresponding author at: Potsdam Institute for Climate Impact Research, Potsdam, Germany.
** Corresponding author.
E-mail addresses: zhen.su@pik-potsdam.de (Z. Su), meyerhenke@hu-berlin.de (H. Meyerhenke).

https://doi.org/10.1016/j.ins.2024.121096
Received 6 March 2023; Received in revised form 7 June 2024; Accepted 22 June 2024

Available online 28 June 2024

ones (e.g., in social networks or web graphs). Moreover, one can construct *functional networks* by statistically relating one time series to another. For example, in climate science, climate data are represented in the form of networks for the spatiotemporal pattern analysis of the underlying climate system [3]: by (1) viewing grid points (or locations) on Earth as nodes and (2) establishing edges between nodes according to the similarity or causality between the corresponding time series at the pair of nodes.

Analyzing large networks in their entirety can be difficult due to the rapid growth of the typical data volume. For example, millions of tweets per day are generated online in social networks; when dealing with climate data, taking the original spatial resolution (e.g., $0.25° \times 0.25°$) can lead to a network with more than half a million nodes and half a billion edges [4]. In such cases, carrying out non-local structural queries or visualizations is time-consuming or even prohibitive. One common solution in the literature is lossy compression, i.e., to discard a large proportion of possibly redundant edges by sparsification to obtain a sparse subgraph $\hat{G}$ of the input graph $\mathcal{G} = (V, E)$. To retain the contextual meaning of nodes, we focus on sparsification and do not consider graph coarsening where nodes can be aggregated. Under the basic premise of preserving essential network properties, sparsification allows a faster and sometimes even more accurate analysis of the available network data [5–7].

Which properties to preserve during sparsification, depends on the application context. Among other properties, theoretical work considers spectral properties such as eigenvalues [8]. Preserving spectral properties is time-consuming, as it requires the solution of many Laplacian linear systems. In practice, alternative objectives that can be computed faster are often preferred [9]. For this, a body of fast edge sampling methods have been proposed, including probability-based [10–12] and filtering-based [13–15] edge sampling (see Section 2). These sampling methods are edge-focused and depend heavily on an appropriate way to define edge importance, especially filtering-based methods.

To relax such a dependency, we propose a generalized edge sampling method with a node-focused perspective. Node-focused refers here to relying on local node characteristics rather than characterizing edge importance. Such an idea is motivated by the fact that local structural characteristics can define not only the basic but also the global organization of a network, as indicated in Refs. [16,17]. The following four applications provide supporting evidence:

- *Random graph generation* [16,17]. In a deterministic graph like $\mathcal{G}$, each edge exists with probability 1. For $\mathcal{G}$, one can generate random graphs reproducing complex/non-local properties of $\mathcal{G}$, by preserving subgraph-based distributions of $\mathcal{G}$, e.g., degree distribution, degree correlations, clustering, and so on.
- *Uncertain graph sampling* [18–20]. In an uncertain graph $\mathcal{G}^U$, each edge exists with probability in $(0, 1]$, and a *possible world* is a deterministic subgraph drawn from the distribution defined by $\mathcal{G}^U$. For $\mathcal{G}^U$, one can extract representative possible worlds by preserving the expected degrees and the expected number of triangles associated with nodes of $\mathcal{G}^U$. The extracted representatives preserve the structural properties of $\mathcal{G}^U$, facilitating statistically sound uncertain graph queries.
- *Degree-based edge sampling* [21,22]. By preserving the expected degrees of nodes, one can obtain sparse subgraphs $\hat{G}$ that preserve the degree distribution of $\mathcal{G}$ well.
- *Centrality correlation* [23,24]. For general networks, different centralities are correlated with each other. This is also true for the degrees, the number of triangles, and the number of non-closed wedges associated with nodes. Besides, these three local properties are interdependent by the definition of the local clustering coefficient [25].

Therefore, in the context of sparsification, it is natural to ask what local node characteristics look like in the sparsified subgraphs. The applications *uncertain graph sampling* [18–20] and *degree-based edge sampling* [21,22] indicate the direction. Specifically, we preserve not only the expected degrees, but also the expected number of triangles and the expected number of non-closed wedges, which are all measures associated with nodes. In particular, we take into account the preservation of triangles due to triangles being able to discern abnormal (sub)graphs and nodes in social networks [26], and to reveal the hidden thematic structure in the World Wide Web [27]. Note that we particularly separate non-closed wedges from closed wedges – the latter are triangles. As we will see later, our proposed approach implicitly preserves basic but also complex network properties.

To the best of our knowledge, limited work is closely related to our objective. Zeng et al. [21,22] formulate a similar approach as an optimization problem (see *degree-based edge sampling* mentioned above). In their work, sparse subgraphs, which are generated by preserving the expected degrees of nodes, preserve implicitly network properties, such as degree distribution and shortest-path distance distribution. However, they only consider the preservation of the expected degrees – which seems overly myopic. Our objective is a generalized version in comparison, since we consider additionally the expected number of triangles and the expected number of non-closed wedges.

From a technical point of view, we transfer the results from *uncertain graph sampling* [19] to network sparsification; we do so by adapting the game-theoretic sampling proposed by Ref. [19]. Our rationale for doing so is: (i) *degree-based edge sampling* [21,22] can be technically regarded as a special case of *uncertain graph sampling*, in a sense that a uniform probability is assigned to each edge of $\mathcal{G}^U$; (ii) our objective generalizes *degree-based edge sampling*, and preserves similar local node properties to those in *uncertain graph sampling*; (iii) game-theoretic sampling performs best for *uncertain graph sampling* while guaranteeing convergence.

The paper is organized as follows. Section 2 reviews the related work on edge sampling and exact potential games. The proposed edge sampling method and algorithms are explained in Sections 3 and 4, respectively. Section 5 presents experimental evaluations, and Section 6 discusses the *tri-wedge assortativity* for selecting algorithms when sparsifying a given graph. Finally, Section 7 concludes this paper.

1.1. Contributions

For graph sparsification, we propose an edge sampling method with a different perspective, which is motivated by *uncertain graph sampling* [19] and *degree-based edge sampling* [21,22]. From a technical viewpoint, we adapt game-theoretic sampling method from *uncertain graph sampling* to graph sparsification. Our contributions are:

- We propose a generalized node-focused edge sampling framework by preserving the expected local properties, including the expected degrees, the expected number of triangles, and the expected number of non-closed wedges, associated with nodes in the sparse subgraphs.
- By including a tolerance threshold into the game-theoretic sampling adapted from *uncertain graph sampling* [19], we significantly accelerate the convergence, while maintaining the sparsification quality.
- Our proposed method maintains a better overall similarity of the sparse subgraphs to the original graphs than other state-of-the-art sampling methods.
- We provide guidance for how to select one of the algorithmic variants involved in our sparsification framework, i.e., the game-theoretic sparsification with tolerance (GST) and further with triangle-emphasis (GSTT), when applying them to an unknown graph.

Compared to the conference version of this paper [1], which considers only functional climate networks, this extended version aims to make our proposed method as general as possible for network data from various domains. To this end, we not only propose additional optimization objectives, but also provide comprehensive studies using additional evaluation metrics and one more state-of-the-art sampling method. The highlights of this extension include: (i) a constant time complexity for computing the expected number of non-closed wedges, assuming that the degree and the number of triangles have been computed already; (ii) an additional algorithmic variant GSTT as part of our framework, for which we relax the independence assumption imposed on the computation of the expected number of triangles; (iii) a measure called *tri-wedge assortativity* that helps to decide which of the algorithms from our framework to apply to an unknown graph.

2. Related work

We review first edge sampling methods, including probability-based and filtering-based ones. We also provide the necessary background on exact potential games, since we adapt a game-theoretic sampling method [19] in our work.

Probability-based edge sampling. One way to sample edges from a given graph $\mathcal{G}$ is according to some probability distribution. The most intuitive one is uniform sampling [11], which preserves spectral properties (i.e., the Laplacian eigenvalue distribution of the generated sparse subgraph $\hat{G}$ is similar to that of $\mathcal{G}$) with high probability, but only over smooth inputs [10]. Spectral properties such as eigenvalue distributions are important in the context of sparsification, due to their capability to characterize graph topology and dynamical processes on networks [28]. Le [12] studied one non-uniform sampling approach. It assigns a probability to each edge $e = \{u, v\}$ that is inversely proportional to the number of common neighbors of the two nodes u and v. This method is similar to uniform sampling if the number of common neighbors becomes small, or similar to effective-resistance-based sampling [8] if the number of common neighbors is large. Effective resistance, on the other hand, stems from viewing a graph as a resistive circuit, where an edge of weight w becomes a resistor of resistance $\frac{1}{w}$. Intuitively, it is low if two vertices are connected via many paths of short length. Using probabilities based on effective resistance leads to sampling methods with strong guarantees on the sparsified subgraph. Generally, strong guarantees require algorithms that are rather time-consuming in practice. Spectral sparsification, for example, requires the solution of numerous linear systems [8].

Filtering-based edge sampling. There is a body of sampling methods interested in identifying important edges (often referred to as edge centrality) [13] to preserve structural properties, e.g., community structure [29,14,15] or the largest connected component [13]. In particular, Hamann et al. [13] have compared systematically several well-known filtering-based edge sampling methods, such as using Jaccard similarity [14], Simmelian backbones [30], and algebraic distance [31]. The general sampling process used therein contains two primary steps: edge scoring and filtering. Edge scoring assigns each edge a value to characterize the importance of that edge; filtering then removes all edges with scores below a certain threshold such that the network is compressed to a desired ratio. They also proposed a sampling method named *local degree*. It preserves the largest connected component of the original graph well in the sparse subgraph.

Both probability-based and filtering-based edge sampling methods listed here can be implemented using only local information without access to the entire network. Therefore, as we also see from our extensive experimental comparisons in Section 5, they are fast and scale to very large instances.

Background on exact potential games. A strategic game $\langle P, \{S_p\}_{p\in P}, \{C_p(S_p, S_{-p})\}_{p\in P} \to \mathbb{R}\rangle$ is a triplet which consists of players p, the strategy S_p of a player p, and the individual cost defined by a cost function C_p. Given initialized strategies for all players, the game proceeds in a round-robin fashion. In every round, each player p minimizes its cost C_p based on the strategies S_{-p} of all other players. Such a process is called *best-response dynamics* [32]. Noth that p changes its current strategy S_p to a new one S'_p if and only if (iff) the gain $g(p)$ is positive, i.e., iff $g(p) = C_p(S_p, S_{-p}) - C_p(S'_p, S_{-p}) > 0$. A strategic game has a *(pure) Nash equilibrium* if the game terminates; that is, no player has the incentive to change its current strategy. A strategic game is called a *potential game* if there is a single global function – the potential function Φ – that represents the incentives of all players to change their strategies. Furthermore, a potential game is said to be *exact* if the gain in the cost function is reflected in the potential function, i.e.,

Fig. 1. An example of computing the expected degree of node A, the expected number of triangles, and the expected number of non-closed wedges associated with A. (a) The given graph $\mathcal{G}$. The degree of A, the number of triangles, and the number of non-closed wedges associated with A, are $m_2^A(\mathcal{G}) = 4$, $m_3^A(\mathcal{G}) = 2$ (i.e., $\{\{A,E\},\{A,D\},\{E,D\}\}$ and $\{\{A,B\},\{A,C\},\{B,C\}\}$), and $m_w^A(\mathcal{G}) = 4$ (i.e., $\{\{A,E\},\{A,C\}\}$, $\{\{A,E\},\{A,B\}\}$, $\{\{A,D\},\{A,C\}\}$, and $\{\{A,D\},\{A,B\}\}$), respectively. (b) The given graph $\mathcal{G}$ with a uniform and independent sampling probability $p = 0.7$. Based on Eqs. (2a), (2b), and (2c), the expected degree of A, the expected number of triangles, and the expected number of non-closed wedges associated with A, are $\mathbb{E}_2^A = 2.8$, $\mathbb{E}_3^A = 0.686$, and $\mathbb{E}_w^A = 2.254$, respectively.

$C_p(S_p, S_{-p}) - C_p(S'_p, S_{-p}) = \Phi(S_p, S_{-p}) - \Phi(S'_p, S_{-p})$. More importantly, the theory of best-response dynamics on an exact potential game guarantees its convergence to a Nash equilibrium, regardless of the initialization [32].

3. Problem definition

3.1. Preliminaries

Let $\mathcal{G} = (V, E)$ be an undirected and unweighted simple (without self-loops, without multi-edges) graph, where V is the set of nodes and $E \subseteq V \times V$ is the set of edges. The goal of graph sparsification is to find a subgraph $\hat{G} = (V, \hat{E})$ that preserves certain network properties of $\mathcal{G}$ in a scaled manner, i.e., where preservation measures reflect the different numbers of edges. Both $\mathcal{G}$ and $\hat{G}$ have the same number of nodes as we do not consider node aggregation. For sparsification, we assume a uniform and independent sampling probability $p \in (0, 1]$ for each edge. We use this probability p to derive the expected local node properties, because our objective is to preserve scaled local node characteristics. The most common symbols used throughout this work are shown in Table 1.

3.2. Sparsification via scaled local properties

For graph sparsification, Zeng et al. [21,22] take only the expected local degrees into account. To expand on their approach, we further consider 3-size subgraphs and propose a *normalized* definition of network sparsification. Although the preservation of expected local node properties can be extended to subgraphs of larger size, the computational cost can be prohibitive if considering subgraphs with more than three nodes [19,33]. The expected degrees, the expected number of triangles (i.e., closed wedges), and the expected number of non-closed wedges associated with nodes have been defined in the context of *uncertain graph sampling* [19] and can be applied here as well.

For a node i of a given $\mathcal{G}$ represented by its adjacency matrix $\mathbf{A}$, the degree of i, the number of triangles, and the number of non-closed wedges associated with i are defined as:

$$m_2^i(\mathcal{G}) := \sum_{j=1}^{|V|} \mathbf{A}_{ij} \tag{1a}$$

$$m_3^i(\mathcal{G}) := \frac{1}{2} \sum_{j=1}^{|V|} \sum_{k=1}^{|V|} \mathbf{A}_{ij}\mathbf{A}_{ik}\mathbf{A}_{jk} \tag{1b}$$

$$m_w^i(\mathcal{G}) := \frac{1}{2} m_2^i(m_2^i - 1) - m_3^i \tag{1c}$$

We can now define the expected local node properties, based on the given sampling probability p. Using the independence assumption as in Ref. [19] and the linearity of expectation, we define the expected degree of i, the expected number of triangles, and the expected number of non-closed wedges associated with i as:

$$\mathbb{E}_2^i := p m_2^i(\mathcal{G}) \tag{2a}$$

$$\mathbb{E}_3^i := p^3 m_3^i(\mathcal{G}) \tag{2b}$$

$$\mathbb{E}_w^i := \frac{1}{2} p^2 m_2^i(\mathcal{G})(m_2^i(\mathcal{G}) - 1) - \mathbb{E}_3^i, \tag{2c}$$

where the independence assumption is applied in Eqs. (2b) and (2c) (see Fig. 1 for an example). More precisely, each triangle and each non-closed wedge is assumed to be preserved with the probability p^3 or p^2, respectively. Therefore, in Eq. (2c), the term $\frac{1}{2} p^2 m_2^i(\mathcal{G})(m_2^i(\mathcal{G}) - 1)$ represents the expected number of wedges (including both closed and non-closed ones) to be preserved in the sparsified graph $\hat{G}$. Note that the conference version of this paper [1] uses a recursive dynamic-programming algorithm [34] to compute the expected number of non-closed wedges $\mathbb{E}_w^i$. Here, by exploiting Eq. (2c), we simplify the computational effort to constant time, given that $m_2^i(\mathcal{G})$ and $m_3^i(\mathcal{G})$ have been computed by Eqs. (1a) and (1b), respectively.

In the desired sparse subgraph $\hat{G}$, each node should be as close as possible to its expected local properties. For this, we define the *normalized* node-level and graph-level distances between $\hat{G}$ and the current subgraph G', which are motivated by Ref. [19].

Table 1
List of symbols.

Symbol	Definition
$\mathcal{G}=(V,E)$	An undirected and unweighted graph with V and E as vertex and edge sets, respectively
$\mathbf{A}$	The unweighted adjacency matrix of $\mathcal{G}$
p	The uniform and independent sampling probability $p\in(0,1]$
$\mathbb{E}_2^i$, $\mathbb{E}_3^i$, $\mathbb{E}_w^i$	The expected degree ($\mathbb{E}_2^i$) of node i, and the expected number of triangles ($\mathbb{E}_3^i$) and the expected number of non-closed wedges ($\mathbb{E}_w^i$) associated with the node i, based on $\mathcal{G}$ and p
$G'=(V,E')$	The current sparse subgraph during edge sampling with V and E' as vertex and edge sets, respectively
$G^*=(V,E^*)$	The final sparse subgraph output by our proposed method with V and E^* as vertex and edge sets, respectively
$\hat{G}=(V,\hat{E})$	The desired sparse subgraph V and $\hat{E}$ as vertex and edge sets, respectively
$\mathbb{E}_3$, $\mathbb{E}_w$	The overall expected number of triangles ($\mathbb{E}_3$) and the expected number of non-closed wedges ($\mathbb{E}_w$), based on $\hat{G}$
$m_3(\cdot)$, $m_w(\cdot)$	The overall number of triangles ($m_3(\cdot)$) and the overall number of non-closed wedges ($m_w(\cdot)$) of a given graph; for example, $m_3(\mathcal{G})$ and $m_w(\mathcal{G})$ are for $\mathcal{G}$, while $m_3(G')$ and $m_w(G')$ are for G'
$m_2^i(\cdot)$, $m_3^i(\cdot)$, $m_w^i(\cdot)$	The degree ($m_2^i(\cdot)$) of node i, and the number of triangles ($m_3^i(\cdot)$) and the number of non-closed wedges ($m_w^i(\cdot)$) associated with the node i, based on a given graph; for example, $m_2^i(\mathcal{G})$, $m_3^i(\mathcal{G})$, and $m_w^i(\mathcal{G})$ are for $\mathcal{G}$, while $m_2^i(G')$, $m_3^i(G')$, and $m_w^i(G')$ are for G'
$\Delta_{2,3,w}^i(\cdot)$	The node-level distance of node i of a subgraph of $\mathcal{G}$ to its expectations, when preserving degrees, triangles, and non-closed wedges (with subscripts '2', '3', and 'w', respectively) in expectations; for example, $\Delta_{2,3,w}^i(G')$ is for G', while $\Delta_{2,3,w}^i(G^*)$ is for G^*
$\Delta_{2,3,w}(\cdot)$	The graph-level distance of a subgraph of $\mathcal{G}$ to the desired sparse subgraph $\hat{G}$, , when preserving degrees, triangles, and non-closed wedges; for example, $\Delta_{2,3,w}(G')$ is for G', while $\delta_{2,3,w}(G^*)$ is for G^*
$\mathrm{GST}_{2,3,w}$	The proposed algorithm: game-theoretic sparsification with tolerance, by default, preserving degrees, triangles, and non-closed wedges in expectation; the preservation of subset properties leads to GST_2, GST_3, and $\mathrm{GST}_{2,3}$
$\mathrm{GSTT}_{2,3,w}$	The proposed algorithmic variant as an extension: game-theoretic sparsification with tolerance and with triangle-emphasis, by default, preserving degrees, triangles, and non-closed wedges in expectation; the preservation of subset properties leads to GSTT_2, GSTT_3, and $\mathrm{GSTT}_{2,3}$

Definition 1. Given the expected degree $\mathbb{E}_2^i$ of node i, the expected number of triangles $\mathbb{E}_3^i$ and the expected number of non-closed wedges $\mathbb{E}_w^i$ associated with i, and the current subgraph $G'\subseteq\mathcal{G}$, the node-level distance of i to its overall expectation is:

$$\Delta_{2,3,w}^i(G') := \frac{1}{m_2^i(\mathcal{G})}|m_2^i(G')-\mathbb{E}_2^i| + \frac{1}{m_3^i(\mathcal{G})}|m_3^i(G')-\mathbb{E}_3^i| + \frac{1}{m_w^i(\mathcal{G})}|m_w^i(G')-\mathbb{E}_w^i|, \tag{3}$$

where $m_2^i(\mathcal{G})$ is the degree of node i based on the given graph $\mathcal{G}$; $m_3^i(\mathcal{G})$ and $m_w^i(\mathcal{G})$ are the number of triangles and the number of non-closed wedges associated with i, respectively. Similarly, $m_2^i(G')$, $m_3^i(G')$, and $m_w^i(G')$ are corresponding values based on G'. Therefore, $\frac{1}{m_2^i(\mathcal{G})}$, $\frac{1}{m_3^i(\mathcal{G})}$, and $\frac{1}{m_w^i(\mathcal{G})}$ are values fixed for a particular graph and precomputable. We use them as normalization factors to avoid the domination of any single absolute value in Eq. (3). Specifically, taking high-degree nodes as an example, during sparsification, their degrees increase or decrease due to the preservation or removal of edges, respectively. This can further lead to quite large changes regarding the number of triangles or non-closed wedges associated with these nodes. Without normalization, such changes would dominate Eq. (3). Note that previous studies [19,21,22] ignore this factor, but we demonstrated its importance in the conference version of this paper [1].

The graph-level distance for a subgraph G' to its overall expectation is therefore defined by following Ref. [19] as:

Definition 2. Given the current subgraph $G'\subseteq\mathcal{G}$, the graph-level distance of G' to its overall expectation is:

$$\Delta_{2,3,w}(G') := \sum_{i\in V}\Delta_{2,3,w}^i(G') \tag{4}$$

The network sparsification problem via scaled-local-property-based edge sampling is therefore defined as:

Definition 3. (Sparsification via scaled local properties). Given an undirected and unweighted graph $\mathcal{G}=(V,E)$ and a uniform and independent sampling probability $p\in(0,1]$, find a sparsified subgraph $\hat{G}=(V,\hat{E})$ such that:

$$\hat{G} := \operatorname*{argmin}_{G'\subseteq\mathcal{G}}\Delta_{2,3,w}(G'), \tag{5}$$

where the sampling probability p is used for defining $\Delta_{2,3,w}(G')$ (see Eqs. (2a), (2b), (2c), (3), and (4)). As specified by Eq. (5), this is meant as the argmin for all three properties together (see $\mathrm{GST}_{2,3,w}$ and $\mathrm{GSTT}_{2,3,w}$ in Table 1 and in Section 4). In our experiments, however, we also look at subsets thereof, i.e., degrees and triangles; that is, $\operatorname*{argmin}_{G'\subseteq\mathcal{G}}\left[\frac{1}{m_2^i(\mathcal{G})}|m_2^i(G')-\mathbb{E}_2^i| + \frac{1}{m_3^i(\mathcal{G})}|m_3^i(G')-\mathbb{E}_3^i|\right]$ (see

$GST_{2,3}$ and $GSTT_{2,3}$ in Table 1 and in Section 4). According to Ref. [18], when preserving only the expected degrees, this problem is a special case of the closest vector problem, which is $\mathcal{NP}$-hard [35]. As our problem is a generalization, it is $\mathcal{NP}$-hard, too. The decision variant is $\mathcal{NP}$-complete: it is easy to see that verification of a solution takes only polynomial time. As no exact polynomial-time algorithm is known, we aim to provide heuristic solutions that are fast and accurate enough for practical purposes.

3.3. Emphasizing on the expected number of triangles

The sparse subgraph $\hat{G}$ by Eq. (5) depends on how to appropriately derive the expected number of triangles $\mathbb{E}_3^i$ and the expected number of non-closed wedges $\mathbb{E}_w^i$ associated with node i. By default, $\mathbb{E}_3^i$ and $\mathbb{E}_w^i$ are derived using the independence assumption. However, considering the diversity of network data and potential edge dependencies [36], the independence assumption cannot always be suitable. Therefore, we consider the case when the assumption of independence is relaxed as an algorithmic variant (see Table 1 and Section 4 for GSTT with the second 'T' representing triangle-emphasis). This corresponds to the second highlighted extension of this paper (see Section 1.1).

A conservative way to implement this idea is to emphasize only the expected number of triangles $\mathbb{E}_3^i$ to be preserved in the sparsified graph $\hat{G}$. That is, we use Eq. (6) in replacement of Eq. (2b) for this purpose, leading to GSTT (see Algorithm 1):

$$\frac{1}{2}(p^3 + p)m_3^i(\mathcal{G}), \tag{6}$$

where the probability that a triangle is preserved in $\hat{G}$ is in $[p^3, p]$, with p^3 being the lower bound based on the independence assumption and p the upper bound. For example, in Fig. 1(b), the probability that the triangle $\{\{A,B\},\{A,C\},\{B,C\}\}$ is preserved is in $[0.343, 0.7]$; we then take the mean 0.5215 for calculating the expected number of triangles based on Eq. (6). Our rationale for doing so is that the triangle is a more important motif structure than the non-closed wedge, because it affects the structure and functionality of a graph to a larger extent [26,27]. In this way, we also adjust the ratio between the overall expected number of triangles $\mathbb{E}_3 := \sum_{i\in V}\mathbb{E}_3^i$ and the overall expected number of non-closed wedges $\mathbb{E}_w := \sum_{i\in V}\mathbb{E}_w^i$ to be preserved in $\hat{G}$, as they are tied with each other by Eq. (2c). Our empirical studies indicate the benefit of considering edge dependencies, particularly in observed real-world networks.

When (not) to consider the relaxation of the independence assumption is investigated in Section 6. We conjecture that it depends on the ratio between the overall number of triangles $m_3(\mathcal{G})$ and the overall number of non-closed wedges $m_w(\mathcal{G})$ in the given graph $\mathcal{G}$:

$$m_3(\mathcal{G}) := \sum_{i\in V} m_3^i(\mathcal{G}) \tag{7a}$$

$$m_w(\mathcal{G}) := \sum_{i\in V} m_w^i(\mathcal{G}) \tag{7b}$$

Specifically, if $\mathcal{G}$ has fewer triangles than non-closed wedges, i.e., $m_3(\mathcal{G}) < m_w(\mathcal{G})$, one needs to avoid a too rapid decline of $\mathbb{E}_3$, as such a decline would lead to more dissimilarities between $\mathcal{G}$ and $\hat{G}$. Therefore, relaxing the independence assumption by Eq. (6) to increase $\mathbb{E}_3$ is necessary. Similarly, if $m_3(\mathcal{G}) > m_w(\mathcal{G})$ for $\mathcal{G}$, one needs to increase $\mathbb{E}_w$, and this can be numerically achieved by retaining the independence assumption by Eq. (2b).

4. Game-theoretic sparsification algorithms: with tolerance (GST) and with triangle-emphasis (GSTT)

Parchas et al. [19] proposed a sampling method based on game theory for *uncertain graph sampling*. It constitutes an exact potential game with convergence to a Nash equilibrium [32]. We adapt this framework to sparsification for two main reasons (see Section 1): (i) *degree-based edge sampling* [21,22] can be technically regarded as a special case of *uncertain graph sampling*; (ii) our objective generalizes the *degree-based edge sampling*. The convergence of the game-theoretic sampling allows us to include a tolerance threshold T for faster convergence (see Fig. 5 as an example).

In the context of sparsification, each edge $e = \{u, v\} \in E$ in a given $\mathcal{G}$ is modeled as a player involved in a strategic game. Each edge has the same binary strategies: 0 for removal and 1 for preservation. The objective of each edge is to make the current subgraph G' as close as possible to the desired subgraph $\hat{G}$ by decreasing the distance of the current local properties of each node to their expectations, as given in Eq. (5). Therefore, there exists a global function $\Phi = \sum_{i\in V}\Delta_{2,3,w}^i(G')$ (same as Eq. (4)) to ensure that such a strategic game is also a potential game. Meanwhile, a change of the strategy of e affects a limited number of nodes locally near e in terms of their current local properties in G', since we consider in our objectives subgraphs with up to 3 nodes. Let the set of nodes affected by a strategy change of $e = (u, v)$ be denoted by $A(e) \subseteq V$. If e alters its strategy, the degrees of u and v change; the number of triangles and the number of non-closed wedges associated with them may change as well. Similar changes apply to the common neighbors of u and v. Thus, $A(e)$ consists of u, v, and the common neighbors of u and v in G'. Consequently, each edge has its own cost function $C_e = \sum_{i\in A(e)}\Delta_{2,3,w}^i(G')$ to minimize.

When the gain in the individual cost induced by the strategy update of e is reflected in the gain in the potential function, a potential game is called exact. Parchas et al. [19] have proved that their game-theoretic sampling method for *uncertain graph sampling* constitutes an exact potential game. This still holds in our case, although we consider also non-closed wedges in our optimization objective. Suppose that the decision of e changes the current $G' = (V, E')$ into $G'' = (V, E'')$; the *gain* $g(e)$ in the potential function Φ induced by a strategy change of e is defined as (following Ref. [19]):

Algorithm 1: Game-theoretic sparsification with tolerance (GST) and with triangle-emphasis (GSTT).

Input: An undirected and unweighted graph $\mathcal{G} = (V, E)$, a uniform and independent sampling probability $p \in (0, 1]$, and the tolerance threshold $T = 0.01$.
Output: $G^* = (V, E^*)$

```
                                                    // Stage I (The expected basic properties)
1  for i ∈ V do in parallel
2      Compute m_2^i(G), m_3^i(G), m_w^i(G), E_2^i, E_3^i, and E_w^i based on Eqs. (1a), (1b), (1c), (2a), (2b) or (6), and (2c)
                                                    // Note that Eqs. (2b) and (6) yield GST and GSTT, respectively
                                                    // Stage II (Sparsification)
3  G' ← G
4  for i ∈ V do in parallel
5      m_2^i(G') ← m_2^i(G); m_3^i(G') ← m_3^i(G); m_w^i(G') ← m_w^i(G)
6  L_new ← V; Potential[|V|] ← 0; r ← 0
7  repeat
8      L ← L_new; L_new ← ∅
9      for each e = {u, v} ∈ E incident (in G) to a node in L do
10         A(e) ← {u} ∪ {v} ∪ {z ∈ V : {z, u} ∈ E' ∧ {z, v} ∈ E'}
11         Compute g'(e) based on Eq. (9)
12         if g'(e) > 0 then
13             if e ∈ E' then
14                 E' ← E' \ {e}
15             else
16                 E' ← E' ∪ {e}
17             Update m_2^i(G'), m_3^i(G'), and m_w^i(G'), based on G'
18             L_new ← L_new ∪ A(e)
19     r ← r + 1; Potential[r] ← (1/|V|) Δ_{2,3,w}(G')
20 until r ≥ 2 and Potential[r − 1] − Potential[r] ≤ T
21 return G* ← G'
```

$$g(e) := \sum_{i \in V} (\Delta^i_{2,3,w}(G') - \Delta^i_{2,3,w}(G'')), \tag{8}$$

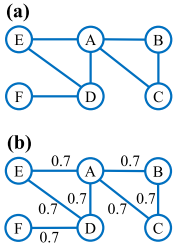
where $\sum_{i \in V} \Delta^i_{2,3,w}(G')$ is the potential of e by retaining its current strategy, while $\sum_{i \in V} \Delta^i_{2,3,w}(G'')$ is the potential of e by changing its strategy. A positive gain with $g(e) > 0$ is desirable, because it means that G'' is closer to our objective than the current subgraph G'. Similarly, the gain in the cost function C_e is:

$$g'(e) := \sum_{i \in A(e)} (\Delta^i_{2,3,w}(G') - \Delta^i_{2,3,w}(G'')). \tag{9}$$

Note that Eq. (8) is actually equivalent to Eq. (9) due to $\forall i \in V \setminus A(e)$: $\Delta^i_{2,3,w}(G') = \Delta^i_{2,3,w}(G'')$; this equivalence ensures the existence of an exact potential game and allows for faster updates. The best-response dynamics – that each edge repeatedly changes its strategy (i.e., preservation or removal) to minimize its cost based on the decisions of all others – in the exact potential game guarantees the convergence to a Nash equilibrium [32]. That is, if the corresponding algorithm models this process, it will terminate.

We implement the game-theoretic sparsification framework with tolerance (GST) and further with triangle-emphasis (GSTT). Algorithm 1 presents the pseudocode of GST/GSTT, which models an exact potential game. The inputs include an undirected and unweighted graph $\mathcal{G}$ and two scalars, i.e., a uniform and independent sampling probability $p \in (0, 1]$ for sparsification and the tolerance threshold T for early termination. Stage I (lines 1-2) computes the degrees of nodes, the number of triangles, the number of non-closed wedges, and their expectations, based on $\mathcal{G}$ and p. The computations can be parallelized easily in several ways, since they need only local information. Stage II initializes the current subgraph G' with $\mathcal{G}$ in line 3. The values $m^i_2(G')$, $m^i_3(G')$, and $m^i_w(G')$ in line 5 are therefore exactly the same as $m^i_2(\mathcal{G})$, $m^i_3(\mathcal{G})$, and $m^i_w(\mathcal{G})$, respectively. L_{new} represents the set of all affected nodes and is initialized with the entire set V. We include another array $Potential[|V|]$ for recording $\frac{1}{|V|}\Delta_{2,3,w}(G')$ during iterations. Starting from line 7, the algorithm proceeds in rounds. In each round, given an edge e incident to a node in L, it first finds all nodes in G' affected by the decision of e. That is, $A(e)$ includes u, v, and common neighbors of u and v in G'. Then, it computes $g'(e)$ induced by an assumed change in the state of e. If $e \in E'$ and the removal of e leads to a positive $g(e)$, then e changes from 1 to 0. If $e \notin E'$ and the preservation of e gives a positive $g(e)$, then e switches from 0 to 1. An example is given in Fig. 2. The iteration stops when the gain progress is smaller than the threshold T. Empirical studies on functional, observed real-world, and synthetic networks indicate that $T = 0.01$ is a good choice in practice for faster convergence while maintaining the quality of the sparse subgraphs.

Time complexity. Regarding (sequential) time complexity, we note for Stage I that computing Eq. (2a) takes $\mathcal{O}(|E|)$ time. When computing triangles, we use a merge-based intersection operation between u and each of its neighbors, since each node already has a sorted neighbor set. Computing Eq. (2b) therefore takes $\mathcal{O}(d_{max}|E|)$ time in total, where $d_{max} = \max\{m^i_2(\mathcal{G}) : i \in V\}$ is the maximum degree in $\mathcal{G}$. (According to [37], an even tighter bound is $\mathcal{O}(a(\mathcal{G})|E|)$, with $a(\mathcal{G})$ being the arboricity of $\mathcal{G}$.) Computing the expected number of non-closed wedges $\mathbb{E}^i_w$ for each node by Eq. (2c) takes constant time. Hence, the total time complexity of Stage I is $\mathcal{O}(d_{max}|E|)$. Stage II depends mostly on the time spent on the repeat-loop. Finding $A(e)$ involves a linear-time intersection operation



	$m_2^i(\mathcal{G})$	$m_3^i(\mathcal{G})$	$m_w^i(\mathcal{G})$	$\mathbb{E}_2^i$	$\mathbb{E}_3^i$	$\mathbb{E}_w^i$
A	4	2	4	2.8	0.686	2.254
B	2	1	0	1.4	0.343	0.147
C	2	1	0	1.4	0.343	0.147
D	3	1	2	2.1	0.343	1.127
E	2	1	0	1.4	0.343	0.147
F	1	0	0	0.7	0	0

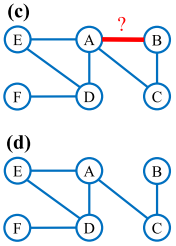


Fig. 2. An example of graph sparsification using $\text{GST}_{2,3,w}$ (see Table 1 and Section 4). (a) The given graph $\mathcal{G}$. (b) The given graph $\mathcal{G}$ with a uniform and independent sampling probability $p = 0.7$. The table lists the degrees of nodes, the number of triangles and the number of non-closed wedges associated with nodes, and their expectations, based on $\mathcal{G}$ and p (see Eqs. (1a), (1b), (1c), (2a), (2b), and (2c)). (c) After initializing G' with $\mathcal{G}$ (see line 6 of Algorithm 1), $\text{GST}_{2,3,w}$ is considering at round 1 whether the edge $e = \{A, B\}$ should still be preserved in G'. If e retains its current strategy of preservation, the gain (or the distance from G' to the overall expectation by Eq. (4)) is 5.958; if it chooses removal from G', yielding a new subgraph G'' as (d), the gain becomes 4.107. The gain $g(e) = 5.958 - 4.107 = 1.851$ is positive. Thus, e is removed and the current subgraph G' becomes (d).

with $\mathcal{O}(d_{max})$ time each for two already sorted neighbor sets. For similar reasons as in Stage I, the for-loop takes $\mathcal{O}(d_{max}|E|)$ time (per iteration of the repeat-loop). Stage II therefore needs $\mathcal{O}(rd_{max}|E|)$ in total, where r is the number of iterations of the repeat-loop. Thus, in total, the time complexity of Algorithm 1 is $\mathcal{O}(rd_{max}|E|)$. We show in Section 5.2 how the tolerance threshold T affects the empirical convergence positively. Moreover, we present in Section 5.4 the empirical average running times of the whole GST/GSTT.

Before moving forward to the next part, it is worth emphasizing two important points. First, we consider by default the preservation of all three expected local node properties in GST/GSTT, leading to algorithms $\text{GST}_{2,3,w}$/$\text{GSTT}_{2,3,w}$. Empirical studies still need to compare them with the case where the preservation of the expected number of non-closed wedges is not considered, i.e., $\text{GST}_{2,3}$ and $\text{GSTT}_{2,3}$. Second, we mentioned in Section 3.2 the case where the dependencies between edges in a triangle subgraph should be taken into account as GSTT. GSTT emphasizes the overall expected number of triangles, particularly for a given graph $\mathcal{G}$ with fewer triangles than non-closed wedges, i.e., $m_3(\mathcal{G}) < m_w(\mathcal{G})$. The asymptotic time complexity of GSTT matches that of GST; the empirical running time and the sparsification quality is investigated in the following section for all four proposed algorithmic variants: $\text{GST}_{2,3}$, $\text{GST}_{2,3,w}$, $\text{GSTT}_{2,3}$, and $\text{GSTT}_{2,3,w}$.

5. Experimental evaluation

We assess the performance of our GST/GSTT framework by addressing the following questions in Sections 5.2, 5.3, and 5.4, respectively:

Q1: How well does a sparse G^* graph generated by GST/GSTT preserve expected local properties?
Q2: How well does a sparse G^* graph generated by GST/GSTT preserve expected non-local / complex properties?
Q3: What is the empirical running time of GST/GSTT? Can instances with a few million edges be processed in reasonable time?

5.1. Experimental settings

(1) Data sets. Extending the conference version of this paper, we consider more and a wider range of network data sets, including functional, observed real-world, and synthetic networks. All 27 networks are summarized in Table 2.

- *Functional climate networks.* Our interest in climate data is mainly driven by studies of complex climate phenomena using complex networks during the last two decades [3]. The reconstructed functional climate networks can be large, especially when a high spatial resolution is considered. From Glo_ERA5SP to Glo_ERA5WVFVC, the first eight networks are reconstructed from daily ERA5 reanalysis data (available online[1]), within the June-July-August season from 1998 to 2019, with the global spatial resolution of $1° \times 1°$. For these climate data, the functional network reconstruction process is adapted from Ref. [38] by viewing grid points as nodes and using Spearman's correlation (which is applicable because the time series are smooth enough) as the similarity between time series. To define edges, we take pair-wise correlations with a significance level of 0.05 and keep only the entries from the highest 5% of the absolute values. The other two functional climate networks, i.e., Glo_TRMM and ASM_TRMM, use the observational data of global precipitation from the Tropical Rainfall Measuring Mission 3B42v6 product (TRMM).[2] The reconstruction process for these two networks is the same as in Refs. [4,39].
- *Observed real-world networks.* From Chameleon to Twitch, the selected thirteen networks describe social, biological, and technological relationships and are available online from popular repositories.[3,4]

[1] https://cds.climate.copernicus.eu/.
[2] https://disc.gsfc.nasa.gov/datasets/.
[3] http://snap.stanford.edu/.
[4] https://networkrepository.com/index.php.

Table 2
Characteristics of data sets.

Type	Network	Nodes ($\|V\|$)	Edges ($\|E\|$)	$\frac{\|E\|}{\|V\|}$	Description
Functional climate networks	Glo_ERA5SP	7,320	593,736	81.11	Global surface pressure from ERA5 data
	Glo_ERA5ST	7,320	882,102	120.51	Global surface temperature from ERA5
	Glo_ERA5GPH	7,320	778,757	106.39	Global 250-hPa geopotential height from ERA5
	Glo_ERA5OLR	7,320	422,724	57.75	Global outgoing long-wave radiation from ERA5
	Glo_ERA5WUC	7,320	541,877	74.03	Global 250-hPa zonal wind from ERA5
	Glo_ERA5WVC	7,320	340,725	46.55	Global 250-hPa meridional wind from ERA5
	Glo_ERA5WVFUC	7,320	482,762	65.95	Zonal component of global 250-hPa vertically integrated water vapor flux from ERA5
	Glo_ERA5WVFVC	7,320	344,020	47	Meridional component of global 250-hPa vertically integrated water vapor flux from ERA5
	Glo_TRMM	36,000	2,139,214	59.42	Precipitation from TRMM data
	ASM_TRMM	20,000	1,771,609	88.58	Precipitation from TRMM in Asian monsoon
Observed real-world networks	Chameleon	2,277	31,371	13.78	Wikipedia articles on chameleons
	FBEgo	4,039	88,234	21.85	Social circles from Facebook
	Crocodile	11,631	170,773	14.68	Wikipedia articles on crocodiles
	HepPh	12,008	118,489	9.87	Collaboration on Arxiv High Energy Physics
	AstroPh	18,772	198,050	10.55	Collaboration on Arxiv Astro Physics
	ASI	34,761	107,720	3.1	Autonomous systems of the Internet
	Enron	36,692	183,831	5.01	Email communication from Enron
	Livemocha	104,103	2,193,083	21.07	Language learning community from Livemocha
	Squirrel	5,201	198,353	38.14	Wikipedia articles on squirrels
	Worm	16,347	762,822	46.66	Gene functional associations
	Recmv	61,989	2,811,458	45.35	Ratings between users and movies
	Catster	149,700	5,448,197	36.39	Friendships from Catster
	Twitch	168,114	6,797,557	40.43	Mutual followers from Twitch
LFR networks	$\text{LFR}_{\mu=0.1}$	10,000	252,039	25.20	Synthetic benchmark
	$\text{LFR}_{\mu=0.2}$	10,000	252,916	25.29	Synthetic benchmark
	$\text{LFR}_{\mu=0.3}$	10,000	249,388	24.94	Synthetic benchmark
	$\text{LFR}_{\mu=0.4}$	10,000	251,048	25.1	Synthetic benchmark

- *LFR networks.* The last four networks are synthetic, constructed with the Lancichinetti-Fortunato-Radicchi (LFR) benchmark [40] implementation in NetworKit [41,42], a tool suite for network analysis on large-scale graphs. Parameters are given as follows: (i) power-law exponent for the degree distribution: $\tau_1 = -2$; (ii) power-law exponent for the community size distribution: $\tau_2 = -1$; (iii) fraction of inter-community edges: $\mu \in \{0.1, 0.2, 0.3, 0.4\}$; (iv) desired average and maximum degrees: 50 and 250, respectively; (v) minimum and maximum sizes of communities: 25 and 250, respectively.

(2) Baselines. We compare GST with five competing methods as baselines, which include two state-of-the-art and three well-known sampling methods.

- *Two state-of-the-art methods.* The first one consists of two variants, which are in our context called $\mathbf{GST}_2$ and $\mathbf{GSTT}_2$. Both methods can be seen as part of our generic framework, but are based on Refs. [21,22] with the same optimization objectives used therein. Zeng et al. [21,22] studied preserving the expected degree of each node and adapted two approximate methods similar to those in *uncertain graph sampling* [19]. Parchas et al. [19] concluded that among all of the approximate methods they proposed, the game-theoretic sampling method generates better representative possible worlds for their use case. As mentioned, we directly adapt this framework for network sparsification. Note that GST_2 and GSTT_2 preserve only the expected degrees. Therefore, the first comparison (see Section 5.2) is between GST_2/GSTT_2 and our generalization $\text{GST}_{2,3}$/$\text{GST}_{2,3,w}$/$\text{GSTT}_{2,3}$/$\text{GSTT}_{2,3,w}$, where 3 and w denote triangles and non-closed wedges, respectively. The second competitor is proposed by Le [12]. The idea is to sample edges with probability inversely proportional to the number of *common neighbors* (**CN**) between two nodes. When the local connectivity defined in his work is sufficiently strong, the graphs sampled by CN show a strong spectral similarity to the original graph (i.e., regarding the Laplacian eigenvalue distribution).
- *Three well-known methods.* The other three competitors are the well-known sampling methods *local degree* (**LD**) [13], *local Jaccard similarity* (**LJS**) [14], and *random edge* (**RE**) [10] (see Section 5.3). We choose them due to their (mostly empirical) effectiveness in preserving the overall connectivity (by LD), community structure (by LJS), and eigenvalue distribution (by RE), as demonstrated previously for non-functional networks. They have been systematically compared by Hamann et al. [13] and implemented in NetworKit [41,42].

(3) Evaluation metrics and procedure. For **Q1**, we analyze the extent to which the expected degree and the expected number of 3-node subgraphs associated with each node are preserved, even when bearing some loss due to the inclusion of the tolerance threshold T in GST/GSTT. The four measures below are used in Section 5.2:

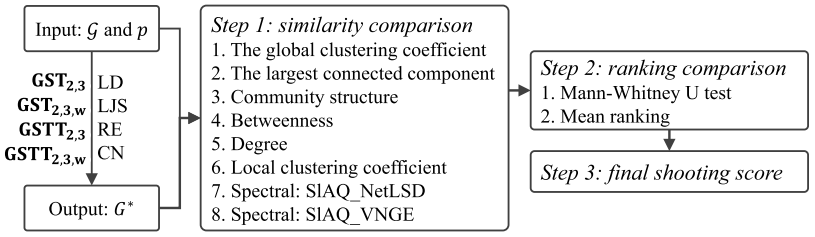


Fig. 3. The workflow for **Q2** (see Section 5.1). For each sampling probability p, Step 1 compares the similarity (regarding different properties) between the original given graph $\mathcal{G}$ and the generated sparse subgraph G^* (see Fig. 6 as an example). Step 2 summarizes rankings among different methods over different properties and p (see Figs. 7, 8, and 9). Step 3 computes the number of data sets for which different methods have relatively better ranking results based on Step 2, over the given 27 networks (see Table 3).

- *Node-wise distance distribution:* $\delta_{2,3,w}(G^*) = \{\Delta^i_{2,3,w}(G^*) : i \in V\}$ and $\delta_{2,3}(G^*) = \{\Delta^i_{2,3}(G^*) : i \in V\}$. $\Delta^i_{2,3}(G^*) := \frac{1}{m^i_2(\mathcal{G})}|m^i_2(G') - \mathbb{E}^i_2| + \frac{1}{m^i_3(\mathcal{G})}|m^i_3(G') - \mathbb{E}^i_3|$ consider only the preservation of degrees and triangles in expectation. $\delta_{2,3,w}(G^*)$ and $\delta_{2,3}(G^*)$ are both sequences of length $|V|$ with each element representing the node-level distance (see Eq. (3) as an example). We visualize $\delta_{2,3,w}(G^*)$ and $\delta_{2,3}(G^*)$ using its distribution.
- *Mean distance:* $\overline{\delta}_{2,3,w}(G^*) = \frac{1}{|V|}\sum_{i\in V}\Delta^i_{2,3,w}(G^*)$ and $\overline{\delta}_{2,3}(G^*) = \frac{1}{|V|}\sum_{i\in V}\Delta^i_{2,3}(G^*)$.
- *Convergence of mean distance:* $\overline{\delta}_{2,3,w}(G') = \frac{1}{|V|}\sum_{i\in V}\Delta^i_{2,3,w}(G')$. This measure is designed for convergence analysis, since it uses the current G' (at line 19 of Algorithm 1) instead of G^*.
- *Cumulative time*: total time spent until the current iteration (lines 7-19 in Algorithm 1), also for empirical convergence analysis.

Regarding **Q2**, a workflow is given in Fig. 3. The core consists of the graph similarity estimation, which assesses to what extent two graphs are similar. In our case, we consider the similarity between the generated sparse subgraph G^* and the original graph $\mathcal{G}$ in terms of the following properties, which cover multiple levels (see Step 1 in Fig. 3): (i) *macroscopic*: the average clustering coefficient and largest connected component; (ii) *mesoscopic*: community structure and betweenness centrality; (iii) *microscopic*: degree and local clustering coefficient; (iv) *spectral*: eigenvalue distribution. These properties have been widely used in various network analysis tasks, including functional climate networks [38,39]. We assume that by preserving these important properties, other structural properties are also preserved to some extent, due to correlations between different properties [23,24]. Computing the exact betweenness values is, in practice, very expensive for the given original network $\mathcal{G}$. Therefore, we use the algorithm EstimateBetweenness [43] implemented in NetworKit [41,42]. Similarly, obtaining the full spectrum of a large graph is computationally prohibitive; therefore, we use the fast approximation techniques SLAQ_NetLSD (based on the heat kernel [44]) and SLAQ_VNGE (based on von Neumann Graph Entropy [45]) proposed by Tsitsulin et al. [46] to compute a low-dimensional representation of a graph. Measures used to estimate the similarity between the properties calculated from G^* and $\mathcal{G}$ are (see Section 5.3):

- Average *Deviation* [13]: we analyze the deviation of the above-mentioned macroscopic properties in G^* from those in $\mathcal{G}$, because these properties are single-valued representations.
- Average *Adjusted rand index* (ARI) [47]: this measure is particularly used for assessing the similarity between two clusterings computed for the final sparse network G^* and the original network $\mathcal{G}$, respectively.
- Average *Spearman's* rank correlation coefficient [13]: microscopic properties are node-wise representations, therefore similarities are estimated using correlation with a significance level of $P < 0.05$. In particular, it is quite likely to have non-significant correlation coefficients, when the sampling probability p decreases to a small value. For such cases, we set the correlation coefficients to be zero; we also avoid as much as possible a small p, especially for networks with a small average degree. In our empirical studies, different ranges of p applied to different networks are given in Table 3.
- Average *Euclidean distance* [44,46]: this measure is also particularly used for the spectral distance comparison, as both G^* and $\mathcal{G}$ are represented by low-dimensional vectors using SLAQ_NetLSD and SLAQ_VNGE. More importantly, the Euclidean distance takes into account the vectors' magnitude, compared with cosine similarity.

The estimation process is as follows. Taking the comparison between $\text{GST}_{2,3}$ and LD as an example, we first generate 10 sparse networks G^* for a given $\mathcal{G}$, based on $\text{GST}_{2,3}$. We choose 10 since all property similarity estimates show a small variance (see Section 5.3). Then, another 10 sparse networks, say LD_{G^*}, are created by using LD with the preservation ratio of edges calculated based on the edge ratio between G^* and $\mathcal{G}$. When estimating the similarity of the community structure, we apply the parallel Louvain method (PLM) [48] from NetworKit [41,42] to $\mathcal{G}$, G^*, and LD_{G^*}, respectively. We then compute the ARI between the highest-quality (out of 100 repeated runs of PLM) community structures obtained from $\mathcal{G}$ and each G^*; the same process is applied to $\mathcal{G}$ and each LD_{G^*}.

One can notice that it is difficult for one edge sampling method to outperform all the others for all these similarity estimations on macroscopic, mesoscopic, and microscopic structural properties. Leskovec and Faloutsos [49] evaluate different algorithms based on different data sets and different evaluation criteria. Partially motivated by their evaluation, we need additional measures to summarize the performance over all of the similarity estimations, instead of checking them one by one (see Step 2 and Step 3 in Fig. 3 and Section 5.3):

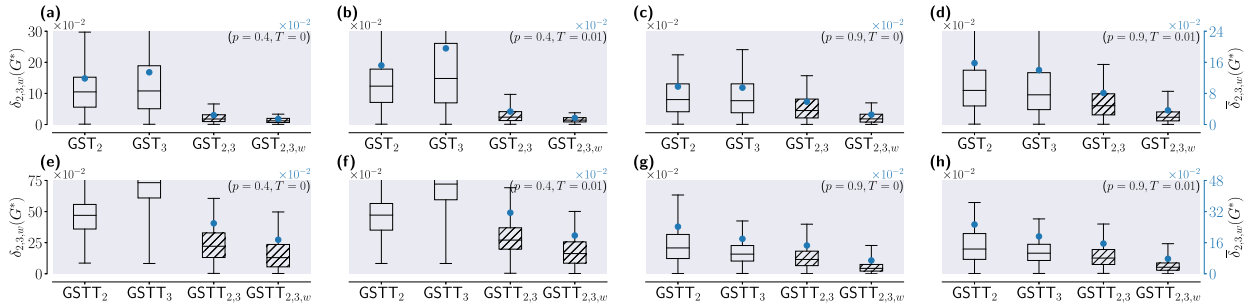


Fig. 4. Node-wise distance distribution (left y-axis, $\delta_{2,3,w}(G^*)$) and mean distance (right y-axis, $\overline{\delta}_{2,3,w}(G^*)$) of the final sparse subgraph G^* to the desired one $\hat{G}$, for Glo_ERA5SP (an example of functional climate networks). On the x-axis are different algorithmic variants under our proposed sparsification framework. Boxplots show how close 0%, 25%, 50%, 75%, and 95% nodes are to their expected local properties. The suffixes of GST/GSTT represent the local properties chosen to be preserved, with '2', '3', and 'w' for degrees, triangles, and non-closed wedges, respectively. The first row is for GST: (a) GST with $p = 0.4, T = 0$; (b) GST with $p = 0.4, T = 0.01$; (e) GST with $p = 0.9, T = 0$; (f) GST with $p = 0.9, T = 0.01$. The second row is for GSTT: (c) GSTT with $p = 0.4, T = 0$; (d) GSTT with $p = 0.4, T = 0.01$; (g) GSTT with $p = 0.9, T = 0$; (h) GSTT with $p = 0.9, T = 0.01$. Figures (a), (b), (e), and (f) produce a sparser structure due to a smaller sampling probability p. These plots illustrate the benefit of preserving both the expected degree of each node *and* the expected number of 3-node subgraphs associated with each node. The hatches indicate the respective best scenarios.

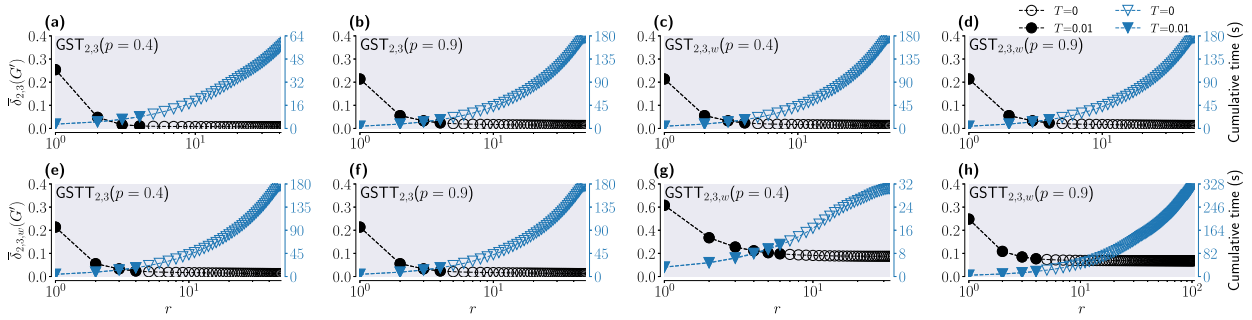


Fig. 5. Convergence of mean distance (left y-axis, $\overline{\delta}_{2,3}(G')$ and $\overline{\delta}_{2,3,w}(G')$) and cumulative time (right y-axis) of GST/GSTT, versus the number of iterations r, for Glo_ERA5SP (an example of functional climate networks). Only $\mathrm{GST}_{2,3}$ and $\mathrm{GST}_{2,3,w}$ are given here since Fig. 4 confirms the better performance when 3-node subgraphs (i.e., triangles and non-closed wedges) are considered for preservation. The first row is for GST: (a) $\mathrm{GST}_{2,3}$ with $p = 0.4$; (b) $\mathrm{GST}_{2,3}$ with $p = 0.9$; (e) $\mathrm{GST}_{2,3,w}$ with $p = 0.4$; (f) $\mathrm{GST}_{2,3,w}$ with $p = 0.9$. The second row is for GSTT: (c) $\mathrm{GSTT}_{2,3}$ with $p = 0.4$; (d) $\mathrm{GSTT}_{2,3}$ with $p = 0.9$; (g) $\mathrm{GSTT}_{2,3,w}$ with $p = 0.4$; (h) $\mathrm{GSTT}_{2,3,w}$ with $p = 0.9$. This figure illustrates the effect of the tolerance threshold $T = 0.01$: it leads to faster convergence (at least 4×, the last blue solid triangle) of GST/GSTT, while nearly retaining the quality of the sparse subgraph obtained with $T = 0$ (the last black solid circle).

- *Ranking distribution*: for each given sampling probability p, each similarity estimation gives a ranking among GST/GSTT, LD, LJS, RE, and CN, from 1 to 5. For each sampling method, we aggregate all rankings over different p and over different similarity estimations on macro-, meso-, and microscopic structural properties. Then, we visualize its ranking distribution for further ranking comparison. The same empirical ranges of p as used for the "average *Spearman* rank correlation coefficient" evaluation above are applied here (also see Table 3).
- *Mean ranking*: the mean of all rankings of each method.
- *Mann-Whitney U test*: The mean ranking is not sufficient to compare the final performance of the five methods, due to the large standard deviation within the ranking distribution. Therefore, instead of directly choosing the one with the smallest mean ranking, we use this U test to classify the five sampling methods (i.e., GST/GSTT, LD, LJS, RE, and CN) into two categories: Group I has better performance, Group II performs worse in comparison. In particular, methods in the two groups satisfy the following conditions: (1) methods in Group I have the same cumulative distribution functions (CDFs) in terms of rankings, given the significance threshold of 0.1; (2) there is at least one method (in Group I) whose CDF is stochastically larger than the CDF of any method in Group II, given the significance threshold of 0.05. Using this test, we can see from empirical studies that Group I also includes the method with the smallest mean ranking, i.e., the best one on average.

For **Q3**, to provide an unbiased comparison between GST/GSTT, LD, LJS, RE, and CN (see Section 5.4), we choose a single-threaded environment without parallelism. We first average the running time over 10 runs (sufficient due to small variance) for each given sampling probability p. Then, by averaging again over different p (see Table 3), we compare the final computational cost for each given graph (see Section 5.4).

5.2. Basic property preservation

We show in detail the distribution of $\delta_{2,3,w}(G^*)$ using boxplots and $\overline{\delta}_{2,3,w}(G^*)$ for Glo_ERA5SP (Fig. 4 in the main text), Chameleon (Fig. S1 in the Supplementary Material), and $\mathrm{LFR}_{\mu=0.1}$ (Fig. S2 in the Supplementary Material), as examples from functional cli-

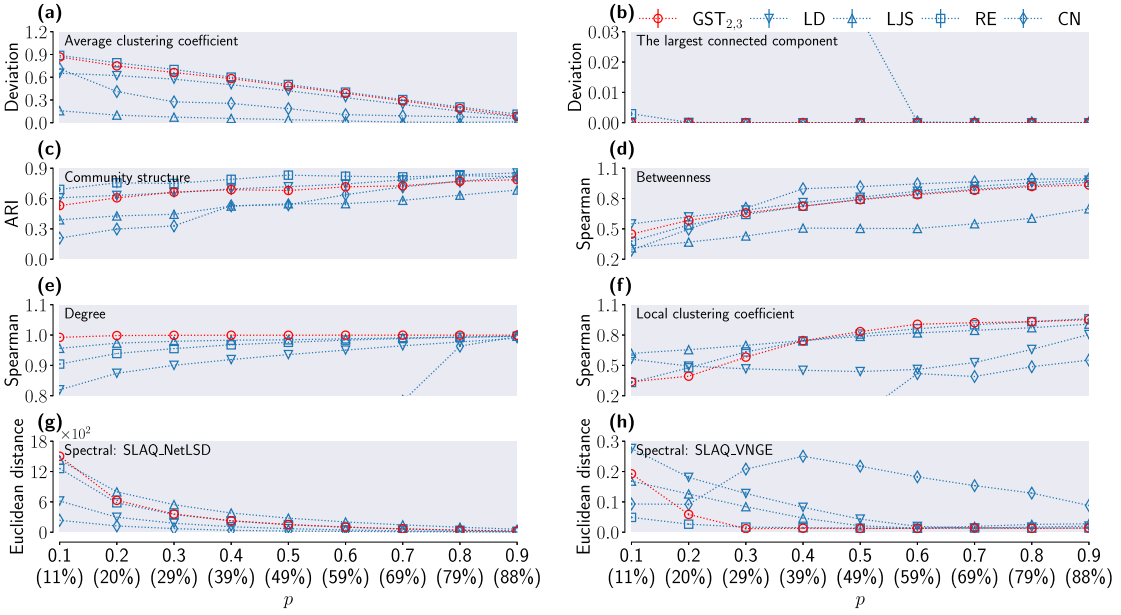


Fig. 6. Comparisons of $GST_{2,3}$, LD, LJS, RE, and CN in preserving eight structural properties, for Glo_ERA5SP (an example of functional climate networks). Each sampling probability p on the x-axis is attached with the exact ratio of preserved edges in brackets. The $GST_{2,3}$ is highlighted in red. This figure indicates that there is no single method that performs better for all of these structural similarity estimations.

mate, observed real-world, and LFR networks. The results for the other networks are similar. The methods with comparably better preservation are highlighted with hatches, all of which involve the preservation of 3-node subgraphs (i.e., triangles and non-closed wedges) for both GST and GSTT. Regarding **Q1**, we conclude that preserving both the expected degrees *and* the expected number of 3-node subgraphs (i.e., $GST_{2,3}$/$GST_{2,3,w}$/$GSTT_{2,3}$/$GSTT_{2,3,w}$) generates sparse structures closer to the expectation than only considering degrees (i.e., GST_2/$GSTT_2$). This fact holds even when the tolerance is set to $T > 0$. From here on we focus only on $GST_{2,3}$/$GST_{2,3,w}$/$GSTT_{2,3}$/$GSTT_{2,3,w}$.

Similarly, for the convergence and cumulative time of GST/GSTT, we also show the results for Glo_ERA5SP (Fig. 5 in the main text), Chameleon (Fig. S3 in the Supplementary Material), and $LFR_{\mu=0.1}$ (Fig. S4 in the Supplementary Material) as examples. Results for the other data sets are again similar to them. When the tolerance parameter is provided (in our case $T = 0.01$), the convergence of $\overline{\delta}_{2,3,w}(G')$ first strictly follows the convergence trajectories with $T = 0$, as to be expected, and then stops due to early termination. The final running times of $GST_{2,3}(T = 0.01)$ and $GST_{2,3,w}(T = 0.01)$ (the last blue solid triangle) are at least 4 times faster than those of $GST_{2,3}(T = 0)$ and $GST_{2,3,w}(T = 0)$ (the last blue hollow triangle), respectively. More importantly, the quality of the respective final sparse structure with $T = 0.01$ (the last black solid circles) is still quite close to those of $T = 0$ (the last black hollow circles). Similar fast convergence can also be observed for $GSTT_{2,3}(T = 0.01)$ and $GSTT_{2,3,w}(T = 0.01)$. Therefore, we use $T = 0.01$ as default value for the following experimental analyses.

5.3. Complex property preservation

As for structural property preservation, how to compare similarity estimates obtained from different structural similarity estimations is not obvious, as mentioned in Section 5.1. We give such a similarity estimation result for Glo_ERA5SP in the main text (see Fig. 6) and similar examples for both observed real-world and LFR networks as Figs. S5 and S6 in the Supplementary Material. For a comprehensive view, we focus on overall rankings in Fig. 7.

As we can see from Fig. 6, $GST_{2,3}$ is quite good at preserving degrees (see Fig. 6(e)), which can be expected due to the explicit preservation of expected degrees. Although Hamann et al. [13] concluded that LD is best for preserving the overall connectivity of a network, we see here from Figs. 6(a) and 6(b) that LJS is even better. This is likely due to different network structures in different domains. They use mostly social networks, while here, Glo_ERA5SP is an instance of the functional climate networks. Another noteworthy point is that for a given sampling probability p, only the similarity estimates of the community structure show a slightly observable variance, while for other estimates, the variance is not visible. This suggests the stability of all these sampling methods applicable to practical scenarios. Still, comparing different structural similarity estimates in Fig. 6 is not conclusive due to the diverse performance of the different methods. We thus summarize Fig. 6 in Fig. 7A(a) by using their rankings. For example, in Fig. 6(e), when $p = 0.1$, the ranking among $GST_{2,3}$, LD, LJS, RE, and CN is $\{1, 4, 2, 3, 5\}$; by enumerating all structural similarity estimations in Fig. 6 over different p, we obtain for each method its ranking results.

Finally, the summarized rankings are presented in Figs. 7, 8, and 9 for functional climate, observed real-world, and LFR networks, respectively. Methods with better relative performance are shown with hatches using the Mann-Whitney U test. We aim to provide a generic sampling method. Therefore, only those algorithmic variants which show consistently better performance in a larger set

Table 3
Summary of the performance of the sampling methods based on ranking distributions in Figs. 7, 8, and 9. Entries marked with ✓ represent methods with relatively better performance.

Type	Network	Sampling probability (p)*	$GST_{2,3}$	$GSTT_{2,3,w}$	LD	LJS	RE	CN	Reference
Functional climate networks	Glo_ERA5SP	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}	✓		✓	✓	✓		Fig. 7A(a)
	Glo_ERA5ST	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}	✓		✓		✓	✓	Fig. 7A(c)
	Glo_ERA5GPH	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}	✓		✓	✓	✓		Fig. 7A(e)
	Glo_ERA5OLR	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}	✓		✓		✓		Fig. 7A(g)
	Glo_ERA5WUC	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}	✓				✓		Fig. 7A(i)
	Glo_ERA5WVC	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}	✓			✓	✓		Fig. 7A(k)
	Glo_ERA5WVFUC	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}	✓				✓		Fig. 7A(m)
	Glo_ERA5WVFVC	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}	✓				✓		Fig. 7A(o)
	Glo_TRMM	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓			✓		Fig. 7B(r)
	ASM_TRMM	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓			✓		Fig. 7B(t)
Observed real-world networks	Chameleon	{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}			✓				Fig. 8B(b)
	FBEgo	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓	✓				Fig. 8B(d)
	Crocodile	{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓	✓				Fig. 8B(f)
	HepPh	{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}						✓	Fig. 8B(h)
	AstroPh	{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓	✓			✓	Fig. 8B(j)
	ASI	{0.7, 0.8, 0.9}		✓		✓	✓	✓	Fig. 8B(l)
	Enron	{0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓	✓	✓		✓	Fig. 8B(n)
	Livemocha	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓		✓			Fig. 8B(p)
	Squirrel	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}			✓				Fig. 8B(r)
	Worm	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓	✓	✓			Fig. 8B(t)
	Recmv	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓	✓	✓	✓		Fig. 8B(v)
	Catster	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓	✓		✓		Fig. 8B(x)
	Twitch	{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓					Fig. 8B(z)
LFR networks	$LFR_{\mu=0.1}$	{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓	✓				Fig. 9B(b)
	$LFR_{\mu=0.2}$	{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓	✓				Fig. 9B(d)
	$LFR_{\mu=0.3}$	{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓					Fig. 9B(f)
	$LFR_{\mu=0.4}$	{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}		✓					Fig. 9B(h)
Summary			$GST_{2,3}$ + $GSTT_{2,3,w}$ = $\frac{24}{27}$		$\frac{15}{27}$	$\frac{8}{27}$	$\frac{13}{27}$	$\frac{5}{27}$	

*The only sampling parameter whose range is empirically selected to avoid non-significant correlations.

of networks are summarized in Table 3. $GST_{2,3}$ works well in both functional climate and LFR networks (see Figs. 7A and 9A). For $GSTT_{2,3,w}$, a consistently better performance in terms of solution quality is observable in the table from Glo_TRMM down to $LFR_{\mu=0.4}$, with the exception of Chameleon, HepPh, and Squirrel (also see Figs. 7B, 8B, and 9B). The performance of $GSTT_{2,3,w}$ is comparable with that of $GST_{2,3}$ in Glo_TRMM, AS_TRMM, and LFR networks. According to the final shooting scores in Table 3, our proposed method (seen as the union of its two variants $GST_{2,3}$ and $GSTT_{2,3,w}$) outperforms the probability-based and filtering-based approaches LD, LJS, RE, and CN; that is, in 24 out of 27 network instances, our proposed method belongs to Group I and has relatively better performance. This answers **Q2**: preserving both expected degrees and 3-node subgraphs yields a sparse subgraph that better preserves complex properties overall. How to choose between $GST_{2,3}$/$GSTT_{2,3,w}$ for sampling an unknown given graph is discussed in Section 6.

5.4. Running times

To answer **Q3**, we compare the running times of $GSTT_{2,3,w}$, LD, LJS, RE, and CN in Fig. 10. According to Table 3, $GSTT_{2,3,w}$ yields the best solution quality, particularly in observed real-world networks. It is also the most time-consuming method, due to the preservation of the expected degrees, the expected number of triangles, and the expected number of non-closed wedges associated with nodes. The running time of $GSTT_{2,3,w}$ thus acts as an upper bound. For a fair comparison, the estimation process is as follows (taking $\mathcal{G}$ = Glo_ERA5SP as an example): for each given sampling probability p, we calculate the average running time when generating 10 sparse subgraphs G^*. The edge ratio between G^* and $\mathcal{G}$ is then used to initialize LD, LJS, RE, and CN, further to obtain their corresponding running times.

According to Ref. [13], the running times of LD and LJS are slightly slower than RE, which only takes linear time in the number of edges. For $GSTT_{2,3,w}$, it depends on the number of iterations r in Stage II, even with the tolerance T included for early termination. In Fig. 10, $GSTT_{2,3,w}$ is roughly 18, 16, 148, and 34 times slower than LD, LJS, RE, and CN, respectively. As for $GST_{2,3}$, it is slightly faster than $GSTTT_{2,3,w}$, with roughly 16, 14, 129, and 28 times slower than LD, LJS, RE, and CN, respectively. Nonetheless, even in the most time-consuming scenario, our proposed method is still applicable to large-scale networks with a few million edges.

6. On the selection of $GST_{2,3}$/$GSTT_{2,3,w}$

Although our framework with $GST_{2,3}$/$GSTT_{2,3,w}$ outperforms LD, LJS, RE, and CN, it remains unclear how to choose the most appropriate algorithm configuration in practice for a given unseen graph. In Table 3, the performance of $GST_{2,3}$ dominates from

(A) $GST_{2,3}$ vs $GST_{2,3,w}$. (B) $GSTT_{2,3}$ vs $GSTT_{2,3,w}$.

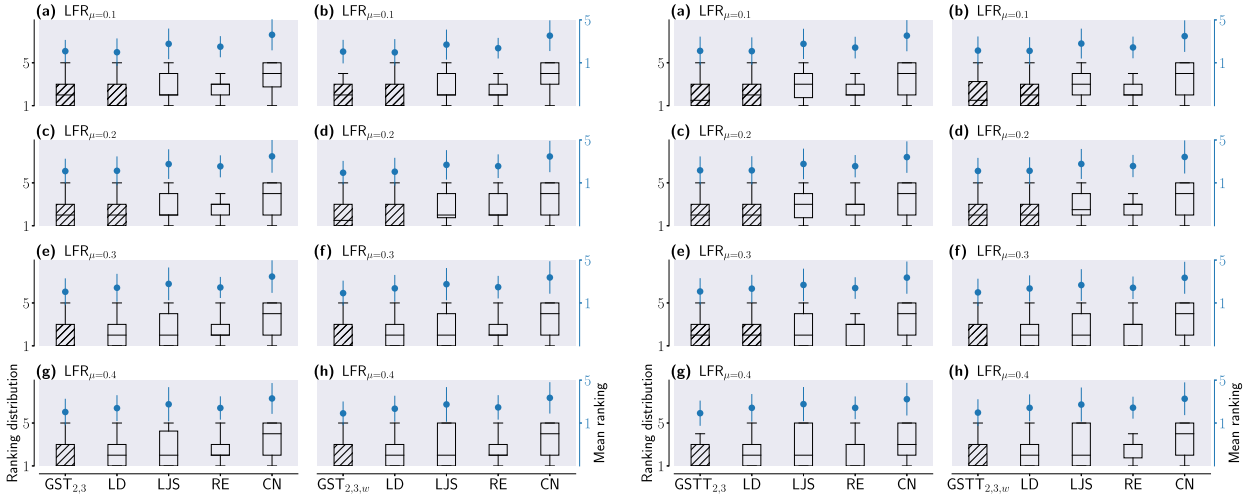
Fig. 7. Ranking comparison of $GST_{2,3}$/$GST_{2,3,w}$/$GSTT_{2,3}$/$GSTT_{2,3,w}$, LD, LJS, RE, and CN, for functional climate networks. Each ranking distribution is summarized over eight structural similarity estimations and over different p (see Fig. 7A(a) as a summarization example for Fig. 6). The considered different sampling probabilities are given in Table 3. For each network, the relatively better sampling methods are highlighted with hatches using the Mann-Whitney U test (see Section 5.1). This figure indicates that the overall performance of $GST_{2,3}$/$GST_{2,3,w}$ by preserving both expected degrees and expected 3-node subgraphs is better than LD, LJS, RE, and CN.

Glo_ERA5SP to Glo_ERA5WVFVC, while from Glo_TRMM down to $LFR_{\mu=0.4}$ (except Chameleon, HepPh, and Squirrel), $GSTT_{2,3,w}$ works consistently well. Finding the graph characteristics that can help with a decision between $GST_{2,3}$ and $GSTT_{2,3,w}$ touches the problem of network representation. In this regard, different network properties are also often considered, in both functional climate and observed real-world networks [38,39]. In our case, network properties that can be computed quickly, especially with access to only local information, are preferable. We use such network properties as prior knowledge for the selection of $GST_{2,3}$ and $GSTT_{2,3,w}$.

Recall that by relaxing the independence assumption, GSTT emphasizes the overall expected number of triangles $\mathbb{E}_3$ to be preserved in the sparse subgraph G^*; consequently, the overall expected number of non-closed wedges $\mathbb{E}_w^i$ is decreased according to Eq. (2c). We have assumed in Section 3.2 that for a given graph $\mathcal{G}$ with fewer triangles than non-closed wedges, GSTT can avoid a rapid decline of $\mathbb{E}_3$. Therefore, we expect that networks from Glo_TRMM down to $LFR_{\mu=0.4}$ (except Chameleon, HepPh, and Squirrel) satisfy $m_3(\mathcal{G}) < m_w(\mathcal{G})$. For this, we define first the node-level difference between triangles and non-closed wedges of $\mathcal{G}$ as:

(A) $GST_{2,3}$ vs $GST_{2,3,w}$. (B) $GSTT_{2,3}$ vs $GSTT_{2,3,w}$.

Fig. 8. Same as Fig. 7, but for observed real-world networks.



(A) GST$_{2,3}$ vs GST$_{2,3,w}$. (B) GSTT$_{2,3}$ vs GSTT$_{2,3,w}$.

Fig. 9. Same as Fig. 7, but for LFR networks.

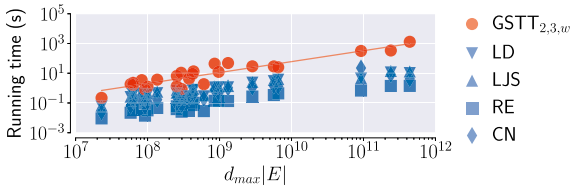


Fig. 10. The running times of GSTT$_{2,3,w}$ (based on Table 3), LD, LJS, RE, and CN, versus $d_{max}|E|$ of 27 networks. GSTT$_{2,3,w}$ is the scenario consuming the most time as it preserves all of the expected local node properties, i.e., degrees, triangles, and non-closed wedges. This figure indicates that our method can be applied to large-scale networks even in the most time-consuming scenario of GSTT$_{2,3,w}$.

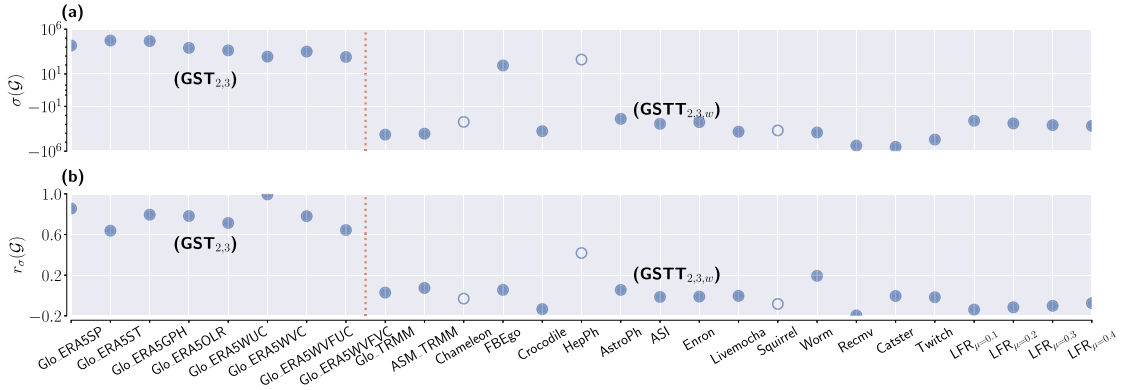


Fig. 11. Network measures for distinguishing 24 out of 27 data sets (without Chameleon, HepPh, and Squirrel) into two groups. Solid blue points are the data sets where our proposed methods show better performance based on Table 3. (a) Graph-level difference between triangles and non-closed wedges $\sigma(\mathcal{G})$ by Eq. (11). (b) *Tri-wedge assortativity* $r_\sigma(\mathcal{G})$ by Eq. (12). This figure indicates that $r_\sigma(\mathcal{G})$ can be a quantitative measure for selecting GST$_{2,3}$ and GSTT$_{2,3,w}$ for an unknown graph.

$$\sigma^i(\mathcal{G}) := m_3^i(\mathcal{G}) - m_w^i(\mathcal{G}) \tag{10}$$

Then the graph-level difference between triangles and non-closed wedges is:

$$\sigma(\mathcal{G}) := \frac{1}{|V|}\sum_{i\in V}\sigma^i(\mathcal{G}) \tag{11}$$

In Fig. 11(a), $\sigma(\mathcal{G}) > 0$ holds from Glo_ERA5SP to Glo_ERA5WVFVC, where GST$_{2,3}$ shows better performance. Also, the reverse observation $\sigma(\mathcal{G}) < 0$ can be made for most networks where GST$_{2,3,w}$ performs better. Still, an exception occurs in the FBEgo network due to $\sigma(\mathcal{G}) > 0$ in Fig. 11(a). One possible reason for this is that $\sigma(\mathcal{G})$ captures only local structure information without considering

the larger connectivity pattern of a given graph $\mathcal{G}$. Therefore, we further define the following measure *tri-wedge assortativity*, based on the definition of *degree assortativity* [50]:

$$r_\sigma(\mathcal{G}) := \frac{\sum_1^{|E|}(Y_i - \overline{Y})(Z_i - \overline{Z})}{\sqrt{\sum_1^{|E|}(Y_i - \overline{Y})^2}\sqrt{\sum_1^{|E|}(Z_i - \overline{Z})^2}} \tag{12}$$

$r_\sigma(\mathcal{G}) \in [-1, 1]$ is essentially the Pearson correlation coefficient and is numerically computed by simply replacing the sequences (i.e., Y and Z) of degree with that of $\sigma^i(\mathcal{G})$. In Fig. 11(b), $r_\sigma(\mathcal{G})$ distinguishes 24 data sets (without Chameleon, HepPh, and Squirrel) into two groups, which are exactly in line with the performance of $\text{GST}_{2,3}$ and $\text{GSTT}_{2,3,w}$ in Table 3. A higher tri-wedge assortativity indicates that nodes with more triangles than non-closed wedges (or those with fewer triangles than non-closed wedges, respectively) tend to connect with each other. For such a graph, our suggestion is to use $\text{GST}_{2,3}$ for sparsification. If $\mathcal{G}$ tends to be lower assortative or even disassortative according to Eq. (12), we suggest to use $\text{GSTT}_{2,3,w}$. From a computational viewpoint, $r_\sigma(\mathcal{G})$ requires the number of triangles $m_3^i(\mathcal{G})$ and the number of non-closed wedges $m_w^i(\mathcal{G})$ associated with a node i. Thus, the computation of $r_\sigma(\mathcal{G})$ can be combined with Stage I of Algorithm 1 at very little extra cost. The tri-wedge assortativity and its analysis constitutes the third highlighted extension of this paper (see Section 1.1).

7. Conclusion

In summary, we proposed a generalized node-focused edge sampling framework for network sparsification. By preserving the expected degrees, the expected number of triangles, and the expected non-closed wedges associated with nodes, complex properties are preserved in the generated sparse subgraph in a self-organized way. Our proposed framework with two algorithmic variants, i.e., $\textbf{GST}_{2,3}$ and $\textbf{GSTT}_{2,3,w}$, generates sparse subgraphs that preserve the overall similarity to the given graph in a considerably better way. Extensive empirical studies verify its better average performance on functional climate, observed real-world, and synthetic LFR networks. We further proposed a network measure, i.e., *tri-wedge assortativity*, which is very effective in guiding the selection between $\textbf{GST}_{2,3}$ and $\textbf{GSTT}_{2,3,w}$.

Regarding future work, one may consider more efficient strategies in refining this node-focused sparsification framework, since our proposed strategy in Section 3.3 is a conservative option. Meanwhile, for practical application purposes, it is also worth investigating how to derive potentially suitable sparsification ratios in advance. An undesired sparsification ratio can lead to too much loss of graph information, and therefore, it should avoided. Besides, whether this sparsification framework works well in weighted graphs is yet to be answered.

CRediT authorship contribution statement

Zhen Su: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Yang Liu:** Writing – review & editing, Methodology, Conceptualization. **Jürgen Kurths:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization. **Henning Meyerhenke:** Writing – review & editing, Validation, Supervision, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

We would like to thank Panos Parchas for data sharing and Anton Tsitsulin for preliminary discussions. Z.S. was funded by the China Scholarship Council (CSC) scholarship. Y.L. was supported by the National Natural Science Foundation of China (Grant No. 62203363). J.K. was supported by the Federal Ministry of Education and Research (BMBF) grant No. 01LP1902J (climXtreme). H.M. was partially supported by German Research Foundation (DFG) grants ME-3619/4-1 (ALMACOM) and GR-5745/1-1 (DyANE). We acknowledge the European Regional Development Fund (ERDF), the German Federal Ministry of Education and Research and the Land Brandenburg for supporting this project by providing resources on the high performance computer system at the Potsdam Institute for Climate Impact Research.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ins.2024.121096.

References

[1] Z. Su, J. Kurths, H. Meyerhenke, Network sparsification via degree- and subgraph-based edge sampling, in: 2022 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2022, pp. 9–16.
[2] M. Newman, Networks, Oxford University Press, 2018.
[3] A.A. Tsonis, P.J. Roebber, The architecture of the climate network, Phys. A, Stat. Mech. Appl. 333 (2004) 497–504.
[4] N. Boers, B. Goswami, A. Rheinwalt, B. Bookhagen, B. Hoskins, J. Kurths, Complex networks reveal global pattern of extreme-rainfall teleconnections, Nature 566 (2019) 373–377.
[5] K. Yanagiya, K. Yamada, Y. Katsuhara, T. Takatani, Y. Tanaka, Edge sampling of graphs based on edge smoothness, in: ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, Singapore, Singapore, 2022, pp. 5932–5936.
[6] M. Choe, J. Yoo, G. Lee, W. Baek, U. Kang, K. Shin, MiDaS: representative sampling from real-world hypergraphs, in: Proceedings of the ACM Web Conference 2022, WWW '22, Association for Computing Machinery, New York, NY, USA, 2022, pp. 1080–1092.
[7] L. Fang, C. Wu, HES: edge sampling for heterogeneous graphs, in: 2023 International Joint Conference on Neural Networks (IJCNN), IEEE, Gold Coast, Australia, 2023, pp. 1–8.
[8] J. Batson, D.A. Spielman, N. Srivastava, S.-H. Teng, Spectral sparsification of graphs: theory and algorithms, Commun. ACM 56 (2013) 87–94.
[9] J. Tětek, M. Thorup, Edge sampling and graph parameter estimation via vertex neighborhood accesses, in: Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing, ACM, Rome Italy, 2022, pp. 1116–1129.
[10] V. Sadhanala, Y.-X. Wang, R. Tibshirani, Graph sparsification approaches for Laplacian smoothing, in: Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, PMLR, 2016, pp. 1250–1259.
[11] J. Lu, H. Wang, Uniform random sampling not recommended for large graph size estimation, Inf. Sci. 421 (2017) 136–153.
[12] C.M. Le, Edge sampling using local network information, J. Mach. Learn. Res. 22 (2021) 1–29.
[13] M. Hamann, G. Lindner, H. Meyerhenke, C.L. Staudt, D. Wagner, Structure-preserving sparsification methods for social networks, Soc. Netw. Anal. Min. 6 (2016) 22.
[14] V. Satuluri, S. Parthasarathy, Y. Ruan, Local graph sparsification for scalable clustering, in: Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data, SIGMOD '11, Association for Computing Machinery, New York, NY, USA, 2011, pp. 721–732.
[15] A. Gionis, P. Rozenshtein, N. Tatti, E. Terzi, Community-aware network sparsification, in: Proceedings of the 2017 SIAM International Conference on Data Mining (SDM), Proceedings, Society for Industrial and Applied Mathematics, 2017, pp. 426–434.
[16] P. Mahadevan, D. Krioukov, K. Fall, A. Vahdat, Systematic topology analysis and generation using degree correlations, ACM SIGCOMM Comput. Commun. Rev. 36 (2006) 135–146.
[17] C. Orsini, M.M. Dankulov, P. Colomer-de-Simón, A. Jamakovic, P. Mahadevan, A. Vahdat, K.E. Bassler, Z. Toroczkai, M. Boguñá, G. Caldarelli, S. Fortunato, D. Krioukov, Quantifying randomness in real networks, Nat. Commun. 6 (2015) 8627.
[18] P. Parchas, F. Gullo, D. Papadias, F. Bonchi, The pursuit of a good possible world: extracting representative instances of uncertain graphs, in: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, SIGMOD '14, Association for Computing Machinery, New York, NY, USA, 2014, pp. 967–978.
[19] P. Parchas, F. Gullo, D. Papadias, F. Bonchi, Uncertain graph processing through representative instances, ACM Trans. Database Syst. 40 (2015) 20:1–20:39.
[20] S. Song, Z. Zou, K. Liu, Triangle-based representative possible worlds of uncertain graphs, in: S.B. Navathe, W. Wu, S. Shekhar, X. Du, S.X. Wang, H. Xiong (Eds.), Database Systems for Advanced Applications, in: Lecture Notes in Computer Science, Springer International Publishing, Cham, 2016, pp. 283–298.
[21] Y. Zeng, C. Song, T. Ge, Selective edge shedding in large graphs under resource constraints, in: 2021 IEEE 37th International Conference on Data Engineering (ICDE), 2021, pp. 2057–2062.
[22] Y. Zeng, C. Song, T. Ge, Y. Zhang, Reduction of large-scale graphs: effective edge shedding at a controllable ratio under resource constraints, Knowl.-Based Syst. 240 (2022) 108126.
[23] D. Schoch, T.W. Valente, U. Brandes, Correlations among centrality indices and a class of uniquely ranked graphs, Soc. Netw. 50 (2017) 46–54.
[24] Z. Su, C. Gao, J. Liu, T. Jia, Z. Wang, J. Kurths, Emergence of nonlinear crossover under epidemic dynamics in heterogeneous networks, Phys. Rev. E 102 (2020) 052311.
[25] D.J. Watts, S.H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (1998) 440–442.
[26] L. Becchetti, P. Boldi, C. Castillo, A. Gionis, Efficient semi-streaming algorithms for local triangle counting in massive graphs, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Las Vegas Nevada USA, 2008, pp. 16–24.
[27] J.-P. Eckmann, E. Moses, Curvature of co-links uncovers hidden thematic layers in the World Wide Web, Proc. Natl. Acad. Sci. 99 (2002) 5825–5829.
[28] P.N. McGraw, M. Menzinger, Laplacian spectra as a diagnostic tool for network structure and dynamics, Phys. Rev. E 77 (2008) 031102.
[29] J. Zhang, K. Zhu, Y. Pei, G. Fletcher, M. Pechenizkiy, Cluster-preserving sampling from fully-dynamic streaming graphs, Inf. Sci. 482 (2019) 279–300.
[30] A. Nocaj, M. Ortmann, U. Brandes, Untangling the hairballs of multi-centered, small-world online social media networks, J. Graph Algorithms Appl. 19 (2015) 595–618.
[31] E. John, I. Safro, Single- and multi-level network sparsification by algebraic distance, J. Complex Netw. 5 (2017) 352–388.
[32] D. Monderer, L.S. Shapley, Potential games, Games Econ. Behav. 14 (1996) 124–143.
[33] J. Nešetřil, S. Poljak, On the complexity of the subgraph problem, Comment. Math. Univ. Carol. 026 (1985) 415–419.
[34] F. Bonchi, F. Gullo, A. Kaltenbrunner, Y. Volkovich, Core decomposition of uncertain graphs, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, Association for Computing Machinery, New York, NY, USA, 2014, pp. 1316–1325.
[35] D. Micciancio, The hardness of the closest vector problem with preprocessing, IEEE Trans. Inf. Theory 47 (2001) 1212–1215.
[36] E.J. Friedman, A.S. Landsberg, J. Owen, W. Hsieh, L. Kam, P. Mukherjee, Edge correlations in spatial networks, J. Complex Netw. 4 (2016) 1–14.
[37] M. Ortmann, U. Brandes, Triangle listing algorithms: back from the diversion, in: 2014 Proceedings of the Meeting on Algorithm Engineering and Experiments (ALENEX), Proceedings, Society for Industrial and Applied Mathematics, 2013, pp. 1–8.
[38] S. Gupta, N. Boers, F. Pappenberger, J. Kurths, Complex network approach for detecting tropical cyclones, Clim. Dyn. 57 (2021) 3355–3364.
[39] Z. Su, H. Meyerhenke, J. Kurths, The climatic interdependence of extreme-rainfall events around the globe, Chaos, Interdiscip. J. Nonlinear Sci. 32 (2022) 043126.
[40] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (2008) 046110.
[41] C.L. Staudt, A. Sazonovs, H. Meyerhenke, NetworKit: a tool suite for large-scale complex network analysis, Netw. Sci. 4 (2016) 508–530.
[42] E. Angriman, A. van der Grinten, M. Hamann, H. Meyerhenke, M. Penschuck, Algorithms for large-scale network analysis and the NetworKit Toolkit, in: H. Bast, C. Korzen, U. Meyer, M. Penschuck (Eds.), Algorithms for Big Data: DFG Priority Program 1736, in: Lecture Notes in Computer Science, Springer Nature, Switzerland, Cham, 2022, pp. 3–20.
[43] R. Geisberger, P. Sanders, D. Schultes, Better approximation of betweenness centrality, in: Proceedings of the Meeting on Algorithm Engineering & Expermiments, Society for Industrial and Applied Mathematics, USA, 2008, pp. 90–100.
[44] A. Tsitsulin, D. Mottin, P. Karras, A. Bronstein, E. Müller, NetLSD: hearing the shape of a graph, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 2347–2356.
[45] P.-Y. Chen, L. Wu, S. Liu, I. Rajapakse, Fast Incremental von Neumann graph entropy computation: theory, algorithm, and applications, in: Proceedings of the 36th International Conference on Machine Learning, PMLR, 2019, pp. 1091–1101.

[46] A. Tsitsulin, M. Munkhoeva, B. Perozzi, Just SLaQ when you approximate: accurate spectral distances for web-scale graphs, in: Proceedings of the Web Conference 2020, WWW '20, Association for Computing Machinery, New York, NY, USA, 2020, pp. 2697–2703.
[47] N.X. Vinh, J. Epps, J. Bailey, Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance, J. Mach. Learn. Res. 11 (2010) 2837–2854.
[48] C.L. Staudt, H. Meyerhenke, Engineering parallel algorithms for community detection in massive networks, IEEE Trans. Parallel Distrib. Syst. 27 (2016) 171–184.
[49] J. Leskovec, C. Faloutsos, Sampling from large graphs, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06, Association for Computing Machinery, New York, NY, USA, 2006, pp. 631–636.
[50] M.E.J. Newman, Assortative mixing in networks, Phys. Rev. Lett. 89 (2002) 208701.