Network Sparsification via Degree- and Subgraph-based Edge Sampling

Zhen Su*†, Jürgen Kurths*‡ and Henning Meyerhenke†

*Potsdam Institute for Climate Impact Research, Potsdam, Germany
†Department of Computer Science, Humboldt-Universität zu Berlin, Berlin, Germany
‡Department of Physics, Humboldt-Universität zu Berlin, Berlin, Germany
Email: zhen.su@pik-potsdam.de, kurths@pik-potsdam.de, meyerhenke@hu-berlin.de

Abstract—**Network (or graph) sparsification compresses a graph by removing inessential edges. By reducing the data volume, it accelerates or even facilitates many downstream analyses. Still, the accuracy of many sparsification methods, with filtering-based edge sampling being the most typical one, heavily relies on an appropriate definition of edge importance. Instead, we propose a different perspective with a generalized local-property-based sampling method, which preserves (scaled) local *node* characteristics. Apart from degrees, these local node characteristics we use are the expected (scaled) number of wedges and triangles a node belongs to. Through such a preservation, main complex structural properties are preserved implicitly. We adapt a game-theoretic framework from uncertain graph sampling by including a threshold for faster convergence (at least 4 times faster empirically) to approximate solutions. Extensive experimental studies on functional climate networks show the effectiveness of this method in preserving macroscopic to mesoscopic and microscopic network structural properties.**

Index Terms—**Graph sparsification, edge sampling, triads**

I. Introduction

Network science facilitates the study of various complex systems. Apart from physically (e.g., technological networks) or conceptually (e.g., social networks) connected entities, time series data from different contexts can be analyzed by relating nodes to each other that are correlated in some way. For example, in climate science, by treating locations on earth as nodes and establishing edges between nodes according to the corresponding time series, climate data are represented as networks [1] (= graphs, we use both terms interchangeably). The resulting objects are often referred to as *functional networks*.

Due to the large size of many real-world networks, downstream analyses, such as visualization and structural queries, can be time-consuming or even prohibitive. A natural solution often seen in the literature is to discard a large proportion of possibly redundant edges by sparsification (without the aggregation of nodes). Under the basic premise of preserving essential network properties, it allows a faster and sometimes even more accurate analysis of the available network data [2].

Which properties to preserve with the subgraph resulting from sparsification, depends on the application context. Theoretical work considered, among others, spectral properties such as eigenvalues [3], requiring the solution of many Laplacian linear systems. For practical applications, alternative objectives that can be computed faster are often preferred. Typically, this happens by sampling the edges to be preserved in the sparser

subgraph G^* according to some probability distribution. The simplest one is uniform sampling, which preserves a type of restricted spectral property with high probability [4]. Other sampling methods aiming at preserving structural properties were systematically compared in [2]. The general sampling process used there contains two primary steps: edge scoring and filtering. Edge scoring assigns each edge a value that describes how 'essential' it is; filtering then removes all edges with scores below a certain threshold such that the network is compressed to a desired ratio.

By preserving degree- and subgraph-based local properties, one can reconstruct complex properties of a given network [5, 6]. This is also true for functional networks. In particular, it has been shown for general networks that a node's importance correlates with its degree as well as with the number of triangles and wedges it belongs to [7]. Motivated by this, we want to sparsify the input graph $\mathcal{G}$ such that these three measures above are preserved in expectation – just scaled appropriately with the sparsification ratio.

To the best of our knowledge, there is limited work closely related to this objective. Zeng et al. [8, 9] formulate a similar approach as an optimization problem, but they preserve only the expected degrees – which seems overly myopic. Since triads (connected 3-node-subgraphs) play an important role in (functional) networks, we thus transfer results from uncertain graph sampling [10, 11, 12] to network sparsification. In uncertain graphs, the objective is to sample a representative instance from the set of all possible instances. We adapt this idea for sampling edges such that the three desired node properties above are retained in expectation. Although the preservation of subgraphs could be extended to larger sizes, the computational cost can be prohibitive above three [11].

The contributions of this paper are as follows: We propose a scaled local-property-based (degrees and 3-node subgraphs) edge sampling, adapted from uncertain graph sampling, for network sparsification. This new perspective of sparsification relaxes the dependency on a specific edge-scoring method. To this end, we adapt a game-theoretic framework [11] and experiments demonstrate that our focus on scaled local properties usually leads to a better preservation of more complex properties than other state-of-the-art sparsification methods.

TABLE I
LIST OF SYMBOLS.

Symbol	Definition
$\mathcal{G} = (V, E, p)$	An undirected network with $|V|$ nodes, $|E|$ edges, and confidence values $p : E \to (0.95, 1]$ associated with the edges
$p^{(S)}$	The generic contribution of each edge to form the final sparse structure, by multiplying a scaling factor $S \in [0, 1]$
$G^* = (V, E^*)$	The final sparse subgraph after edge sampling with $|V|$ nodes, $|E^*|$ edges
$G' = (V, E')$	The current subgraph during edge sampling with $|V|$ nodes, $|E'|$ edges
$l = 2, 3, w$	The basic local properties associated with each node to be preserved, i.e., $l = 2$ for degree, $l = 3$ for triangles, and/or $l = w$ for wedges
$m_l(u, \mathcal{G})$	The possible degree of node u and possible number of triangles and wedges u belongs to based on $\mathcal{G}$
$|L_l(u, \mathcal{G})|$	The maximum possible degree of node u and maximum possible number of triangles and wedges u belongs to based on $\mathcal{G}$
$E[m_l(u, \mathcal{G})]$	The expected degree of node u and expected number of triangles and wedges u belongs to based on $\mathcal{G}$
$m_l(u, G')$	The current degree of node u and current number of triangles and wedges u belongs to based on G'
$\Delta m_l(u, G')$	The distance, for node u, between the current local properties $m_l(u, G')$ based on G' and the corresponding expectations $E[m_l(u, \mathcal{G})]$
$\Delta m_{l=2,3,w}(G')$	The total distance of the current G' to the expectation, summarized over $l = 2, 3, w$ and over all nodes V

II. PROBLEM DEFINITION

A. Preliminaries

Let $\mathcal{G} = (V, E, p)$ be an undirected network, where V is the set of nodes and $E \subseteq V \times V$ is the set of edges. Let $p : E \to (0, 1]$ be an assigned probability to indicate the confidence on the existence of an edge. We consider p particularly for functional networks in this paper, which is why we restrict the range of p to $(0.95, 1]$. They are usually constructed in a statistical manner with high confidences (more details see Section IV-A) from time series. If $p(e) = 1 \ \forall e \in E$, then the constructed network is called deterministic. For the most common symbols used throughout this work, see Table I.

The sparse network $G^* = (V, E^*)$ after edge sampling is a subgraph of $\mathcal{G}$. Both have the same number of nodes as we do not consider node aggregation. To obtain G^* in the desired way, we first need to derive scaled degrees, triangles, and wedges associated with nodes. The basic idea is that each edge in $\mathcal{G}$ contributes to the emergence of observed structural properties, such as degree distribution and community structure. By scaling down the contribution, one can expect the corresponding properties to be scaled similarly. To this end, we include a scaling factor $S \in [0, 1]$ with $p^{(S)}_{e=\{u,v\}} := p(e) \times S \ \forall e \in E$, which implicitly determines the ratio of preserved edges after sparsification.

Since now edges are attached with probabilities, G^* should eventually conform well with the scaled structural properties of $\mathcal{G}$ in *expectation*. That is, we can define the expected degree of a node and the expected number of triangles and wedges the node belongs to, as the scaled local properties to specifically indicate the optimization goal. We also note that the expected number of wedges is often not as important as the (more prominent) expected number of triangles. Therefore, whether to preserve both the expected number of triangles and wedges associated with nodes depends on the application context.

B. Sparsification via scaled local properties

To expand on [8, 9] and to take more than local degrees into account, we further consider 3-size subgraphs and propose a *normalized* definition of network sparsification. The expected

Fig. 1. An example of network sparsification via scaled local-property-based edge sampling in this work. (a) The original undirected network $\mathcal{G}$ with $p^{(S)} = p = 1$ (see Section II-A) as the confidence on the existence of each edge, indicating that each edge fully contributes to the formation of this network. (b) The scaled contribution is assumed to be $p^{(S)} = p \times S = 0.7$. For example, the expected degree of A and the expected number of triangles that A belongs to become $E[m_{l=2}(A, \mathcal{G})] = p^{(S)}_{A,B} + p^{(S)}_{A,C} + p^{(S)}_{A,D} + p^{(S)}_{A,E} = 2.8$ and $E[m_{l=3}(A, \mathcal{G})] = p^{(S)}_{A,B} \times p^{(S)}_{A,C} \times p^{(S)}_{B,C} = 0.343$ (see Section II-B), respectively. (c) The final sparse network G^* obtained by considering the scaled node properties in (b) as the optimization objective (see Section II-B) and by using the adapted heuristic method $GST_{2,3}$ (see Section III).

degree of a node and the expected number of triangles and wedges the node belongs to have been defined in the context of uncertain graphs [11] and can also be applied here.

For a given $\mathcal{G}$ and a randomly selected node u, all possible neighbors of u form the set $L_{l=2}(u, \mathcal{G}) = \{v : \{u, v\} \in E\}$ with $|L_{l=2}(u, \mathcal{G})|$ being the degree of u in $\mathcal{G}$. Given $p^{(S)}$, the corresponding set containing edge contributions is $p^{(S)}_{l=2}(u, \mathcal{G}) = \{p^{(S)}_{u,v} : \{u, v\} \in E\}$. Similarly, all possible triangles that u belongs to form the set $L_{l=3}(u, \mathcal{G}) = \{\{u, v, x\} : \{u, v\}, \{u, x\}, \{v, x\} \in E\}$. The maximum possible number of triangles that u can have in G^* hence is $|L_{l=3}(u, \mathcal{G})|$ and we also have $p^{(S)}_{l=3}(u, \mathcal{G}) = \{\{p^{(S)}_{u,v}, p^{(S)}_{u,x}, p^{(S)}_{v,x}\} : \{u, v\}, \{u, x\}, \{v, x\} \in E\}$.

By assuming the independence of edge probabilities [10], the expected degree of u as well as the expected number of triangles and wedges that u belongs to (respectively), are derived using the linearity of expectation as [11]:

$$E[m_{l=2}(u, \mathcal{G})] := \sum_{i=1}^{|p^{(S)}_{l=2}(u, \mathcal{G})|} p^{(S)}_{l=2}(u, \mathcal{G})_i \qquad (1)$$

$$E[m_{l=3}(u, \mathcal{G})] := \sum_{i=1}^{|p^{(S)}_{l=3}(u, \mathcal{G})|} \prod_{j=1}^{3} p^{(S)}_{l=3}(u, \mathcal{G})_j \qquad (2)$$

$$E[m_{l=w}(u, \mathcal{G})] := \frac{1}{2}\left(E[m_{l=2}(u, \mathcal{G})^2] - E[m_{l=2}(u, \mathcal{G})]\right) \\ - E[m_{l=3}(u, \mathcal{G})] \tag{3}$$

where i and j iterate over the members of the sets $p_{l=2}^{(S)}(u, \mathcal{G})$ and $p_{l=3}^{(S)}(u, \mathcal{G})$, respectively. An example for the calculation of Eqs. (1) and (2) is given in Figure 1b. For Eq. (3), if we let X be a discrete random variable with $X = m_{l=2}(u, \mathcal{G}) = \{x_i | x_i \in [0, |L_{l=2}(u, \mathcal{G})|]\}$, then it represents all possible degrees of u during the edge sampling process. To obtain $E[X^2]$, we calculate $Pr(X = x_i)$ by using dynamic programming based on $p_{l=2}^{(S)}(u, \mathcal{G})$ [13].

In a sparse subgraph G^*, each node should be as close as possible to its expected basic local properties. For this, we define the *normalized* distance from the current degree ($l = 2$) of u and the current number of triangles ($l = 3$) and wedges ($l = w$) that u belongs to in G', to their expectations:

$$\Delta m_l(u, G') := \frac{1}{|L_l(u, \mathcal{G})|}|m_l(u, G') - E[m_l(u, \mathcal{G})]| \tag{4}$$

where $\frac{1}{|L_l(u, \mathcal{G})|}$ is a normalization factor to distinguish the positions of different nodes. It emphasizes that the sparsification by edge sampling is built on top of the original network. Note that previous studies [8, 9, 10, 11] ignore this factor. We demonstrate its importance in Section IV-C. The total distance for a subgraph G' is therefore defined as:

Definition 1. *Given an undirected network $\mathcal{G} = (V, E, p)$ and scaled local properties (on the expected degree of each node and the expected number of triangles and wedges each node belongs to) to be preserved, the distance of any subgraph $G' \subseteq \mathcal{G}$ to its overall expectation is:*

$$\Delta m_{l=2,3,w}(G') := \sum_{u \in V} \sum_{l=2,3,w} \Delta m_l(u, G') \tag{5}$$

The network sparsification problem via edge sampling is therefore defined as:

Definition 2. *(Sparsification via scaled local properties). Given an undirected network $\mathcal{G} = (V, E, p)$, find a subgraph $G^* = (V, E^*)$ such that:*

$$G^* := \underset{G' \subseteq \mathcal{G}}{\operatorname{argmin}} \Delta m_{l=2,3,w}(G') \tag{6}$$

By default, this is meant as the argmin for all three properties together. In our experiments, we will also look at subsets thereof (degrees and triangles), though. According to Ref. [10], for $l = 2$ this problem is a special case of the closest vector problem, which is $\mathcal{NP}$-hard [14]. As our problem is a generalization, it is $\mathcal{NP}$-hard, too. We hence aim at providing heuristic solutions that are fast and accurate enough for practical purposes.

III. THE GAME-THEORETIC SPARSIFICATION WITH TOLERANCE (GST)

Parchas et al. [11] proposed a game-theoretic framework for uncertain graph sampling. It consists of an exact potential game with convergence to a Nash equilibrium based on best-response dynamics [15]. We adapt this framework to sparsification and include a tolerance factor that allows to terminate when the progress is below a user-specified threshold.

The basic idea is that each edge $e = \{x, y\} \in E$ in a given $\mathcal{G}$ is modeled as a player involved in an exact potential game. In this game, the gain change in the individual cost function is reflected in a global potential function. Specifically, each edge decides whether to be preserved (binary states: 1 for preservation) in the final sparse graph G^*. Suppose that the decision of e changes the current $G' = (V, E')$ into $G'' = (V, E'')$; the current state of e is updated only if this leads to a positive *gain* change ($g(e) > 0$), which is defined as [11]:

$$g(e) := \sum_{u \in V} \sum_{l=2,3,w} (\Delta m_l(u, G') - \Delta m_l(u, G'')), \tag{7}$$

where the sum of $\Delta m_l(u, G')$ is the gain (or the distance to the overall expectation) of e by retaining its current state, while the sum of $\Delta m_l(u, G'')$ is the gain of e by changing its state. The result is the overall gain change resulting from the global potential function where each node $u \in V$ is involved. Recall that we consider basic local properties: degrees and 3-node subgraphs (wedges and triangles). Only a limited number of nodes are therefore affected by a change in e in Eq. (7). These nodes include x, y, and common neighbors of x and y in G', forming a set we denote as $L(e)$. Therefore, as proved in [11], Eq. (7) is equivalent to:

$$g(e) := \sum_{v \in L(e)} \sum_{l=2,3,w} (\Delta m_l(v, G') - \Delta m_l(v, G'')) \tag{8}$$

which represents the change of the individual cost function. The equivalence of Eqs. (7) and (8) ensures that this game is an exact potential game. The best-response dynamics – that each edge repeatedly changes its state based on the decisions of all others – on the exact potential game guarantees the convergence to a Nash equilibrium [15]. That is, if the corresponding algorithm models this process, it will terminate.

Algorithm 1 presents the pseudocode of GST, which models such an exact potential game. We emphasize again that although we consider by default the preservation of the expected number of wedges ($l = w$) in GST, empirical studies still need to compare two cases: with and without $l = w$. The inputs include an undirected network $\mathcal{G}$ and two important values, the tolerance T for early termination and the scaling factor S for sparsification. Stage I (lines 1-5) computes the expected local properties based on $\mathcal{G}$ and S. The computation of Eqs. (1) and (2) are parallelized since they need only local information. Stage II initializes the current subgraph G' with the entire set E in line 6. The $m_l(u, G')$ in line 8 are therefore exactly the same as $|L_l(u, \mathcal{G})|$ for $l = 2, 3, w$. L_{new} represents the set of all affected nodes and is initialized with the entire set V. We include another array $Gain[|V|]$ for recording $\Delta m_{l=2,3,w}(G')$ during iterations. Starting from line 10, the algorithm proceeds in rounds. In each round, given an edge e incident to a node in L, it first finds all affected nodes in G' by the decision of e. That is, $L(e)$ includes x, y, and common neighbors of x and

Algorithm 1: Game-theoretic sparsification with tolerance (GST)

Input: An undirected network $\mathcal{G} = (V, E, p)$, tolerance factor $T = 0.01$, scaling factor $S \in [0, 1]$
Output: $G^* = (V, E^*)$
    // Stage I (The expected basic properties)
1   $p^{(S)} \leftarrow p \times S$
2   **for** $u \in V$ **do in parallel**
3   Compute Eqs. (1) and (2), $|L_{l=2}(u, \mathcal{G})|$, and $|L_{l=3}(u, \mathcal{G})|$
4   **for each** $u \in V$ **do**
5   Compute Eq. (3) and $|L_{l=w}(u, \mathcal{G})|$
              // Stage II (Sparsification)
6   $E' \leftarrow E$
7   **for** $u \in V$ **do in parallel**
8      $m_l(u, G') \leftarrow |L_l(u, \mathcal{G})|$ ($l = 2, 3, w$, respectively)
9   $L_{new} \leftarrow V; Gain[|V|] \leftarrow 0; r \leftarrow 0$
10   **repeat**
11      $L \leftarrow L_{new}; L_{new} \leftarrow \emptyset$
12   **for each** $e = \{x, y\} \in E$ *incident (in $\mathcal{G}$) to a node in L* **do**
13          $L(e) \leftarrow \{x, y\} \cup \{u \in V : \{u, x\} \in E' \wedge \{u, y\} \in E'\}$
14   Compute $g(e)$ by Eq. (8)
15   **if** $g(e) > 0$ **then**
16   **if** $e \in E'$ **then**
17                $E' \leftarrow E' \setminus \{e\}; L_{new} \leftarrow L_{new} \cup L(e)$
18   **else**
19                $E' \leftarrow E' \cup \{e\}; L_{new} \leftarrow L_{new} \cup L(e)$
20      $r \leftarrow r + 1; Gain[r] \leftarrow \Delta m_{l=2,3,w}(G')$
21   **until** $r \geq 2$ *and* $Gain[r-1] - Gain[r] \leq T$
22   **return** $E^* \leftarrow E'$

y in G'. Then, it computes $g(e)$ induced by assuming that e changes its current state. 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. The iteration stops based on the progress in the gain in relation to the threshold T.

Regarding (sequential) time complexity, we note for Stage I that computing Eq. (1) 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. (2) therefore takes $\mathcal{O}(d_{max}|E|)$ time in total, where $d_{max} = \max\{|L_{l=2}(u, \mathcal{G})| : u \in V\}$ is the maximum (possible) degree in $\mathcal{G}$. According to [16], an even tighter bound is $\mathcal{O}(a(\mathcal{G})|E|)$, with $a(\mathcal{G})$ being the arboricity of $\mathcal{G}$. Computing Eq. (3) via dynamic programming also takes $\mathcal{O}(d_{max}|E|)$ 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 $L(e)$ involves a linear-time intersection operation 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 IV-B how the tolerance threshold T affects the convergence positively. Moreover, we present in Section IV-D the empirical running times of both stages.

IV. APPLICATIONS TO CLIMATE DATA

We assess the performance of GST by answering the following three questions in Secs. IV-B, IV-C, and IV-D, respectively:

Q1: How well does GST generate a sparse G^* preserving scaled local properties?

Q2: How well does GST generate a sparse G^* preserving non-local / complex properties?

Q3: How is the running time of GST?

A. Experimental settings

(1) Climate data sets. Our particular focus on climate data is mainly driven by studies of complex climate phenomena using complex networks during the last two decades. The reconstructed functional climate networks can be large especially when a high spatial resolution is considered. Four functional networks are summarized in Table II. If $p = 1$, then the corresponding networks are completely deterministic:

- Glo_ERA5SP: we use the time series of daily surface level pressure (SP) within the June-July-August season from 1998 to 2019 from ERA5 reanalysis data [17], with the global spatial resolution of $1° \times 1°$. The functional network reconstruction process is adapted from [18] by viewing grid points as nodes and using Spearman correlation as the similarity between time series.
- Glo_ERA5ST: it is the same as Glo_ERA5SP, but using daily surface level temperature (ST) [17].
- Glo_TRMM: we consider the observational data of global precipitation from Tropical Rainfall Measuring Mission 3B42v6 product (TRMM) [19]. Time series represent the daily rainfall sums within the June-July-August season from 1998 to 2019 with a spatial resolution of $1° \times 1°$. As precipitation data are spiking series, we adopt event synchronization (ES) as a nonlinear similarity measure [20]. By treating grid points as nodes, the reconstruction process is the same as in [1].
- ASM_TRMM: it is the same as Glo_TRMM, but focuses on a relatively small region, i.e., the Asian summer monsoon (ASM) region, instead of the global scale.

(2) Baselines. We compare GST with four baselines. Zeng et al. [8, 9] studied preserving the expected degree of each node and adapted two approximate methods similar to those in uncertain graph sampling [11]. Parchas et al. [11] concluded that among all approximate methods they proposed, the game-theoretic framework generates better representative instances.

TABLE II
CHARACTERISTICS OF DATA SETS

Network	Nodes ($|V|$)	edges ($|E|$)	$\frac{|E|}{|V|}$	Edge confidence (p)
Glo_ERA5SP	7,320	593,736	81.11	1
Glo_ERA5ST	7,320	882,102	120.51	1
Glo_TRMM	36,000	2,139,214	59.42	[0.99, 1]
ASM_TRMM	20,000	1,771,609	88.58	[0.99, 1]

We directly adapt this framework for network sparsification. In particular, GST_2 preserves only the scaled degrees and corresponds to [8, 9]. Therefore, the first comparison is between GST_2 and our extensions $GST_{2,3}$/$GST_{2,3,w}$, where 3 and w denote 3-node subgraphs (triangles and wedges, respectively) associated with each node (see Section IV-B). Other three well-known sampling methods are *local degree* (LD) [2], *local jaccard similarity* (LJS) [21], and *random edge* (RE) [4] (see Section IV-C). We choose them due to their effectiveness in preserving the overall connectivity (by LD), community structure (by LJS), and spectral property (by RE), at least in non-functional networks. They have been systematically compared in [2] and implemented in NETWORKIT [22], a tool suite for scalable network analysis.

(3) Evaluation metrics. 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 with the inclusion of a tolerance threshold T in GST. The four measures below are used (see Section IV-B):

- *Node-wise distance distribution*: $\Delta_{2,3,w}(G^*) = \Delta m_{l=2}(u, G^*) + \Delta m_{l=3}(u, G^*) + \Delta m_{l=w}(u, G^*)$. It represents the summarized overall distance of the local properties (degrees, triangles, and wedges) for each node in the generated G^* to the expectation. Hence, $\Delta_{2,3,w}(G^*)$ is a sequence of length $|V|$. The minimization of the sum of $\Delta_{2,3,w}(G^*)$ over all nodes corresponds to the objective of Eq. (6).
- *Mean distance*: $\overline{\Delta}_{2,3,w}(G^*) = \frac{1}{|V|} \sum_1^{|V|} \Delta_{2,3,w}(G^*)$.
- *Convergence of mean distance*: $\overline{\Delta}_{2,3,w}(G') = \frac{1}{|V|}(\Delta m_{l=2}(u, G') + \Delta m_{l=3}(u, G') + \Delta m_{l=w}(u, G'))$. This measure is designed for convergence analysis, since it is based on the current G' (at line 20 of Algorithm 1) instead of G^*.
- *Cumulative time*: total time spent until the current iteration (lines 10-25 in Algorithm 1), also for empirical convergence analysis.

For **Q2**, the following property queries are considered: *macroscopic*: the global clustering coefficient and largest connected component; *mesoscopic*: community structure and betweenness centrality; *microscopic*: degree and local clustering coefficient. Both mesoscopic and microscopic queries have been applied in functional climate network analysis [1, 18]. In particular, computing the exact betweenness values is in practice very expensive for the unsparsified network. Therefore, we use *ApproxBetweenness* from NETWORKIT [22] with a guarantee that the error is no larger than 0.01, with a probability of at least 0.9. Measures used to estimate the similarity between the properties calculated from G^* and $\mathcal{G}$ are (see Section IV-C):

- Average *Deviation* [2]: we analyze the deviation of the macroscopic properties in G^* from those in $\mathcal{G}$, because these properties are single-valued representations.
- Average *Adjusted rand index* (ARI) [23]: this one is particularly used for giving the similarity between two clusterings obtained based on the final sparse network

G^* and the original network $\mathcal{G}$, respectively.

- Average *Spearman* rank correlation coefficient [2]: microscopic properties are node-wise representations, therefore similarities are estimated using correlation with a significance level of $P < 0.01$.

The estimation process is as follows. Taking the comparison between $GST_{2,3}(T = 0.01)$ and LD as an example, we first generate 100 sparse networks G^* for a given $\mathcal{G}$, based on $GST_{2,3}(T = 0.01)$. Then, another 100 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 $\mathcal{G}$. Assuming the query on community structure, we apply the parallel Louvain method [24] from NETWORKIT [22] to $\mathcal{G}$, G^*, and LD_{G^*}, respectively. We then compute the ARI between the highest-quality (out of 100 repeated runs) 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 hard to ensure that one edge sampling method outperforms the rest for all of these property queries. Therefore, we need additional measures summarizing all queries, instead of checking them one by one (see Section IV-C):

- *Ranking distribution*: for each given scaling factor S, each query task gives a ranking between GST, LD, LJS and RE, from 1 to 4. We summarize all rankings of each method for different S and property queries.
- *Mean ranking*: the mean of all rankings of each method.

For **Q3**, to give a fair comparison between GST, LD, LJS, and RE (see Section IV-D), we choose a single-threaded environment without parallelization. The comparison shows the average running time over 100 runs.

B. Basic property preservation

We show the distribution of $\Delta_{2,3,w}(G^*)$ using boxplots and $\overline{\Delta}_{2,3,w}(G^*)$ in Figures 2, 3, 4 and 5. The preservation scenarios which include 3-node subgraphs (triangles and wedges) are highlighted with hatches (see $GST_{2,3}$ and $GST_{2,3,w}$). Regarding **Q1**, we conclude that preserving both the expected degrees *and* the expected number of 3-node subgraphs generates sparse structures closer to the expectation than only considering degrees. This fact holds even when the tolerance is set to $T > 0$. From here on we focus only on $GST_{2,3}$ and $GST_{2,3,w}$.

As for the convergence and cumulative time of GST, due to limited space, we show the result for Glo_ERA5SP as an example in Figure 6; results for the other three networks are similar to this one. When the tolerance $T = 0.01$ is empirically given, the convergence of $\overline{\Delta}_{2,3,w}(G')$ strictly follows the convergence trajectories of $T = 0$, as it should be. The final running times of $GST_{2,3}(T = 0.01)$ and $GST_{2,3,w}(T = 0.01)$ (blue circles) are at least 4 times faster than that of $GST_{2,3}(T = 0)$ and $GST_{2,3,w}(T = 0)$ (blue triangles). More importantly, the qualities of the final sparse structure by $T = 0.01$ (the last black circles) are still quite close to those of $T = 0$ (the last black triangles).

Fig. 2. The distribution (left y-axis) and mean (right y-axis) of $\Delta_{2,3,w}(G^*)$ based on G^*, versus different preservation scenarios for Glo_ERA5SP. Boxplots show how close 0%, 25%, 50%, 75% and 95% nodes are to their expected local properties (2, 3, and w for degrees, triangles, and wedges, respectively). The suffixes of GST represent the local properties chosen to be preserved. (a) GST(S=0.2, $T = 0$). (b) GST(S=0.2, $T = 0.01$). (c) GST(S=0.9, $T = 0$). (d) GST(S=0.9, $T = 0.01$). (a) and (b) produce a sparser structure due to a smaller S. This figure indicates the benefit of preserving both the expected degree of each node *and* the expected number of 3-node subgraphs each node belongs to.

Fig. 3. Same as Figure 2 but for Glo_ERA5ST.

Fig. 4. Same as Figure 2 but for Glo_TRMM.

Fig. 5. Same as Figure 2 but for ASM_TRMM.

Fig. 6. The convergence ($\overline{\Delta}_{2,3,w}(G')$) and cumulative time of GST base on the current G', versus the number of iterations r for Glo_ERA5SP. (a) $\text{GST}_{2,3}$(S=0.2). (b) $\text{GST}_{2,3}$(S=0.9). (c) $\text{GST}_{2,3,w}$(S=0.2). (d) $\text{GST}_{2,3,w}$(S=0.9). Only $\text{GST}_{2,3}$ and $\text{GST}_{2,3,w}$ are given here since Figures 2, 3, 4 and 5 confirm the better performance when 3-node subgraphs (triangles and wedges) are considered for preservation. This figure indicates the inclusion of the tolerance factor $T = 0.01$ (blue circles) facilitates (at least 4 times faster) the convergence of GST while guaranteeing the quality of the final sparse structure close to $T = 0$ (black circles).

Fig. 7. Comparisons between $\text{GST}_{2,3}$($T = 0.01$), LD, LJS and RE on six structural queries for Glo_ERA5SP. Each scaling factor on the x-axis is attached with the exact ratio of preserved edges in brackets. The $\text{GST}_{2,3}$($T = 0.01$) is highlighted in red. This figure indicates that there is no single method that performs better for all of these queries.

Fig. 8. The ranking distribution and mean ranking of GST ($GST_{2,3}(T = 0.01)$ and $GST_{2,3,w}(T = 0.01)$), LD, LJS and RE, summarized over six property queries for four networks. (a) and (b) Glo_ERA5SP. (a) is also the summarized rankings of Figure 7. (c) and (d) Glo_ERA5SP. (e) and (f) Glo_TRMM. (g) and (h) ASM_TRMM. For each network, the best sampling method is highlighted with hatches and the red dash line is the median of the ranking distribution. This figure indicates that the overall performance of GST by preserving both scaled degrees and 3-node subgraphs is better than filtering-based approaches LD, LJS, and RE.

Fig. 9. The ranking distribution and mean ranking of $GST_{2,3}(T = 0.01)$ and $UNGST_{2,3}(T = 0)$ (the unnormalized version by removing $\frac{1}{|L_l(u,\mathcal{G})|}$ from Eq. (4), summarized over six structural queries for Glo_ERA5SP. GST is highlighted with hatches in boxplots and the median of the ranking distribution is shown with red dash lines. This figure indicates the necessity of including a normalization factor for better performance.

C. Complex property preservation

As for property queries, how to compare the similarity estimates obtained for different queries is not obvious, as mentioned in Section IV-A. We therefore give such a similarity result only for Glo_ERA5SP as an example in Figure 7, and focus on overall rankings in Figure 8. In Figure 7, $GST_{2,3}(T = 0.01)$ and $GST_{2,3,w}(T = 0.01)$ are quite good at preserving degrees (see Figure 7c), which can be expected due to the explicit preservation of scaled degrees. Although Hamann et al. [2] concluded that LD is best for preserving the overall connectivity of a network, we here see from Figures 7a and 7b that LJS is even better. This should be due to different network structures in different domains. They use mostly social networks, while we focus on functional climate networks. Another noteworthy point is that for a given scaling factor S, only similarity estimates of community structure show a slightly larger variance. This suggests the stability of all these sampling methods applicable for practical

Fig. 10. The running times of GST, LD, LJS and RE. For each network, GST chooses the best preservation scenario based on Figure 8. (a) Glo_ERA5SP with $GST_{2,3}(T = 0.01)$. (b) Glo_ERA5ST with $GST_{2,3}(T = 0.01)$. (c) Glo_TRMM with $GST_{2,3,w}(T = 0.01)$. (d) ASM_TRMM with $GST_{2,3}(T = 0.01)$. Clearly, Stage II dominates the running time of GST. This figure indicates that in spite of a higher running time, GST can be applied to large-scale networks.

scenarios. Still, comparing different queries in Figure 7 is not conclusive due to the diverse performance of the different methods. We thus summarize Figure 7 in Figure 8a by using their rankings.

In Figure 8a, $GST_{2,3}(T = 0.01)$ ranks first in the comparisons of both median (red dash lines) and mean (blue dots) rankings. From Figure 8, one can conclude that, for a given $\mathcal{G}$, a good performance of $GST_{2,3}(T = 0.01)$ does not guarantee that $GST_{2,3,w}(T = 0.01)$ also has a similarly good performance. For example, $GST_{2,3}(T = 0.01)$ works better for Glo_ERA5SP, Glo_ERA5ST, and ASM_TRMM, while $GST_{2,3,w}(T = 0.01)$ is better for Glo_TRMM and ASM_TRMM. We conjecture that this is due to the diversity of different network structures, as we expected in Secs. II-A and III. Nonetheless, either one of our two methods is always the best. Thus, preserving both scaled degrees and 3-node subgraphs yields a sparser graph that better preserves complex properties overall. This answers **Q2**.

We also compare $GST_{2,3}(T = 0.01)$ with $UNGST_{2,3}(T = 0)$ to verify the necessity of including a normalization factor $\frac{1}{|L_l(u,\mathcal{G})|}$ in Eq. (4), where $UNGST_{2,3}(T = 0)$ is an unnormalized version of GST by removing from Eqs. (4) and (8) this normalization factor. The same estimation process based on the above six property queries is adopted and the summarized rankings are shown in Figure 9. $UNGST_{2,3}(T = 0)$ proceeds until the final convergence instead of early convergence, since $T = 0$. $GST_{2,3}(T = 0.01)$ still generates sparse networks with a higher similarity to the original Glo_ERA5SP properties.

D. Running times

To answer **Q3**, we compare the running times of GST, LD, LJS and RE in Figure 10. Taking $\mathcal{G} = \text{Glo\_ERA5SP}$ as an example, $\text{GST}_{2,3}(T = 0.01)$ is chosen as the sampling method based on Figure 8a. For each given scaling factor S, $\text{GST}_{2,3}(T = 0.01)$ generates 100 sparse subgraphs G^*. We calculate the average and deviation of running time for both stages of Algorithm 1. The edge ratio between G^* and $\mathcal{G}$ is used for the initialization of LD, LJS, and RE, further to obtain the corresponding running time.

According to [2], the running times of LD and LJS are slightly slower than RE, which only takes linear time in the number of edges. The running time of GST mainly depends on the number of iterations r in Stage II, even with a tolerance factor T included for early termination. In Figure 10, GST is therefore roughly 19, 12, and 90 times slower than LD, LJS, and RE, respectively. Nonetheless, GST is still applicable to large-scale networks.

V. Conclusion

In summary, we proposed a different perspective (by preserving scaled local node characteristics) from the general filtering-based sampling methods for network sparsification. Our empirical studies on functional climate networks verify that the proposed method generates sparse subgraphs that preserve the overall similarity to the original network in a considerably better way.

As future work, we will further study the robustness of this method in other network application scenarios as well as for synthetic data. Which preservation of 3-node subgraphs ($l = 2, 3$ or $l = 2, 3, w$) one should choose for best results on a wide range of unknown data sets remains as another issue not fully settled yet.

References

[1] Z. Su, H. Meyerhenke, and J. Kurths, "The climatic interdependence of extreme-rainfall events around the globe," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 32, no. 4, p. 043126, 2022.

[2] M. Hamann, G. Lindner, H. Meyerhenke, C. L. Staudt, and D. Wagner, "Structure-preserving sparsification methods for social networks," *Social Network Analysis and Mining*, vol. 6, no. 1, p. 22, 2016.

[3] J. Batson, D. A. Spielman, N. Srivastava, and S.-H. Teng, "Spectral sparsification of graphs: Theory and algorithms," *Communications of the ACM*, vol. 56, no. 8, pp. 87–94, 2013.

[4] V. Sadhanala, Y.-X. Wang, and 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.

[5] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat, "Systematic topology analysis and generation using degree correlations," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 135–146, 2006.

[6] 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, and D. Krioukov, "Quantifying randomness in real networks," *Nature Communications*, vol. 6, no. 1, p. 8627, 2015.

[7] M. Benzi and C. Klymko, "On the Limiting Behavior of Parameter-Dependent Network Centrality Measures," *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 2, pp. 686–706, 2015.

[8] Y. Zeng, C. Song, and 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.

[9] Y. Zeng, C. Song, T. Ge, and Y. Zhang, "Reduction of large-scale graphs: Effective edge shedding at a controllable ratio under resource constraints," *Knowledge-Based Systems*, vol. 240, p. 108126, 2022.

[10] P. Parchas, F. Gullo, D. Papadias, and 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*, ser. SIGMOD '14. New York, NY, USA: Association for Computing Machinery, 2014, pp. 967–978.

[11] ——, "Uncertain Graph Processing through Representative Instances," *ACM Transactions on Database Systems*, vol. 40, no. 3, pp. 20:1–20:39, 2015.

[12] S. Song, Z. Zou, and K. Liu, "Triangle-Based Representative Possible Worlds of Uncertain Graphs," in *Database Systems for Advanced Applications*, ser. Lecture Notes in Computer Science, S. B. Navathe, W. Wu, S. Shekhar, X. Du, S. X. Wang, and H. Xiong, Eds. Cham: Springer International Publishing, 2016, pp. 283–298.

[13] F. Bonchi, F. Gullo, A. Kaltenbrunner, and Y. Volkovich, "Core decomposition of uncertain graphs," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '14. New York, NY, USA: Association for Computing Machinery, 2014, pp. 1316–1325.

[14] D. Micciancio, "The hardness of the closest vector problem with preprocessing," *IEEE Transactions on Information Theory*, vol. 47, no. 3, pp. 1212–1215, 2001.

[15] D. Monderer and L. S. Shapley, "Potential Games," *Games and Economic Behavior*, vol. 14, no. 1, pp. 124–143, 1996.

[16] M. Ortmann and U. Brandes, "Triangle Listing Algorithms: Back from the Diversion," in *2014 Proceedings of the Meeting on Algorithm Engineering and Experiments (ALENEX)*, ser. Proceedings. Society for Industrial and Applied Mathematics, 2013, pp. 1–8.

[17] H. Hersbach, B. Bell, P. Berrisford, S. Hirahara, A. Horányi, J. Muñoz-Sabater, J. Nicolas, C. Peubey, R. Radu, D. Schepers, A. Simmons, C. Soci, S. Abdalla, X. Abellan, G. Balsamo, P. Bechtold, G. Biavati, J. Bidlot, M. Bonavita, G. De Chiara, P. Dahlgren, D. Dee, M. Diamantakis, R. Dragani, J. Flemming, R. Forbes, M. Fuentes, A. Geer, L. Haimberger, S. Healy, R. J. Hogan, E. Hólm, M. Janisková, S. Keeley, P. Laloyaux, P. Lopez, C. Lupu, G. Radnoti, P. de Rosnay, I. Rozum, F. Vamborg, S. Villaume, and J.-N. Thépaut, "The ERA5 global reanalysis," *Quarterly Journal of the Royal Meteorological Society*, vol. 146, no. 730, pp. 1999–2049, 2020.

[18] S. Gupta, N. Boers, F. Pappenberger, and J. Kurths, "Complex network approach for detecting tropical cyclones," *Climate Dynamics*, vol. 57, no. 11, pp. 3355–3364, 2021.

[19] G. J. Huffman, D. T. Bolvin, E. J. Nelkin, D. B. Wolff, R. F. Adler, G. Gu, Y. Hong, K. P. Bowman, and E. F. Stocker, "The TRMM Multisatellite Precipitation Analysis (TMPA): Quasi-Global, Multiyear, Combined-Sensor Precipitation Estimates at Fine Scales," *Journal of Hydrometeorology*, vol. 8, no. 1, pp. 38–55, 2007.

[20] R. Quian Quiroga, T. Kreuz, and P. Grassberger, "Event synchronization: A simple and fast method to measure synchronicity and time delay patterns," *Physical Review E*, vol. 66, no. 4, p. 041904, 2002.

[21] V. Satuluri, S. Parthasarathy, and Y. Ruan, "Local graph sparsification for scalable clustering," in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '11. New York, NY, USA: Association for Computing Machinery, 2011, pp. 721–732.

[22] C. L. Staudt, A. Sazonovs, and H. Meyerhenke, "NetworKit: A tool suite for large-scale complex network analysis," *Network Science*, vol. 4, no. 4, pp. 508–530, 2016.

[23] N. X. Vinh, J. Epps, and J. Bailey, "Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance," *Journal of Machine Learning Research*, vol. 11, no. 95, pp. 2837–2854, 2010.

[24] C. L. Staudt and H. Meyerhenke, "Engineering Parallel Algorithms for Community Detection in Massive Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 1, pp. 171–184, 2016.