Network Science **9** (S1): S134–S156, 2021
doi:10.1017/nws.2021.2

CAMBRIDGE UNIVERSITY PRESS

ORIGINAL ARTICLE

Sampling methods and estimation of triangle count distributions in large networks

Nelson Antunes[1,2*], Tianjian Guo[3] and Vladas Pipiras[3]

[1]Center for Computational and Stochastic Mathematics, University of Lisbon, Avenida Rovisco Pais 1049-001, Lisbon, Portugal, [2]University of Algarve, Faro, Portugal and [3]Department of Statistics and Operations Research, University of North Carolina, CB 3260, Chapel Hill, NC 27599, USA (e-mails: Tianjian.Guo@mccombs.utexas.edu, pipiras@email.unc.edu)
*Corresponding author. Email: nantunes@ualg.pt

Action Editor: Hocine Cherifi

Abstract

This paper investigates the distributions of triangle counts per vertex and edge, as a means for network description, analysis, model building, and other tasks. The main interest is in estimating these distributions through sampling, especially for large networks. A novel sampling method tailored for the estimation analysis is proposed, with three sampling designs motivated by several network access scenarios. An estimation method based on inversion and an asymptotic method are developed to recover the entire distribution. A single method to estimate the distribution using multiple samples is also considered. Algorithms are presented to sample the network under the various access scenarios. Finally, the estimation methods on synthetic and real-world networks are evaluated in a data study.

Keywords: triangles; random sampling; distribution estimation; inversion approach; asymptotic approach; multiple samples; static and streaming graphs; power laws

1. Introduction

Triangles formed by edges are the most fundamental topological structures of networks. The number of triangles enters network metrics, such as clustering coefficient or transitivity ratio (Newman, 2018), and serves in network description, analysis, and model building. Applications exploiting triangle counts include spam detection (Becchetti et al., 2008), discovering common topics on the web (Eckmann & Moses, 2002), query plan optimization in databases (Bar-Yossef et al., 2002), community detection (Palla et al., 2005), and others.

Triangle counting has also spurred much theoretical research on suitable counting algorithms for various contexts and goals, for example, to scale well with increasing network size, to adapt to streaming data of edges, to work on networks with restricted access, and many others. The recent review paper (Al Hasan & Dave, 2018) describes well the many developments in this research direction. Algorithms based on various sampling schemes have also been studied actively, especially for large networks, with the goal of estimating the number of triangles space efficiently, even if not quite exactly. See, for example, Jha et al. (2015), Stefani et al. (2017), and the references in Al Hasan & Dave (2018).

In this work, our focus is also on sampling methods and triangles in large networks. But we go beyond the number of triangles as in much of the earlier literature and consider instead the distributions of triangle counts, per both vertex (node) and edge. We are interested in estimating these distributions through suitable sampling schemes. Note that our object of study involves a

range of counts (i.e. the number of vertices or edges participating in 0, 1, ... triangles) and not just a single count as the total number of triangles. For this reason perhaps, the considered distributions of triangle counts have thus far received relatively little attention in the literature. The few related exceptions are the works of Stefani et al. (2017) and Lim et al. (2018), which estimate the number of triangles for each vertex in streaming graphs. However, it is yet to be seen whether these estimates translate successfully to obtain the distributions of triangle counts (see Section 2.3 below for further discussion). These distributions are nevertheless informative, showing how triangles are distributed over the network (scattered or concentrated) with respect to vertices and edges. For example, as we discuss below (Section 3.2.3), these distributions might have power-law tails, suggesting the existence of multiple hub-like vertices/edges participating in large numbers of triangles. For instance, in social networks, the number of triangles of a user is used to identify its social role in the network, and provides a metric in assessing the content quality provided by the user.

Our contributions are several fold.

- We propose a new sampling scheme tailored for estimating triangle count distributions per vertex and edge, which we call *vertex-induced edge* (VIE) sampling. Some available classical sampling schemes such as induced and incident subgraph sampling (e.g. Kolaczyk, 2009, Chapter 5.3.1), often do not carry enough information for proper inference about triangle count distributions; other sampling schemes that seem natural for these distributions can be computationally over-expensive (see Section 2.3 below). VIE sampling balances these issues and is also the scheme that is amenable to theoretical analysis.
- We consider three sampling designs of VIE sampling: without replacement, with replacement, and Bernoulli sampling adapted to network access scenarios with full access, restricted access, and streaming edges, respectively.
- We develop two estimation methods from a VIE sample: an *inversion* approach for the bulk of the distribution that is based on a relationship between the true distribution and the sampled distribution and an *asymptotic* approach for the tail of the distribution that involves an asymptotic equivalence between the two distributions. This follows our and others' earlier work on similar problems in other sampling contexts.
- We consider an estimation method based on scaling the number of triangles of the sampled vertices and edges with VIE using *multiple samples*. A similar approach was used to estimate the numbers of triangles per nodes in streaming data for a unique sample (Lim et al., 2018). The distributions of triangle counts per vertex and edge are then obtained from the empirical distributions of the scaled values. The method will be used as a baseline for comparison with the inversion and asymptotic methods.
- We discuss and present algorithms that implement the VIE sampling approach to the situations of full access, restricted access (where we use random walks), and streaming data of edges (where we employ hashing).
- We examine our proposed sampling and estimation methods on simulated and real-world networks.

The rest of the paper is organized as follows. Section 2 presents quantities of interest and our sampling framework. Section 3 concerns estimation approaches based on inversion and asymptotics. Section 4 considers the case of multiple samples collected in parallel. Section 5 details our sampling algorithms. Section 6 includes data applications. Finally, Section 7 concludes and discusses directions for future work.

This work is an extended version of our conference paper (Antunes et al., 2020). We expand here the previously proposed VIE sampling approach by allowing three sampling designs. As alluded to above, these accommodate the main network access scenarios. The inversion and

Table 1. Summary of notation

Notation	Description
$G = (V, E)$	Graph G with set of vertices V and set of edges E
N/M	Number of vertices/edges in G
$T_w/T_{(u,v)}$	Number of triangles of vertex/edge $w/(u, v)$
$T_v\,(j)/T_e(j)$	Total number of vertices/edges with j triangles
$t_v\,(j)/t_e(j)$	Proportion of vertices/edges with j triangles
N_t/M_t	Maximum number of triangles per vertex/edge
V^*/E^*	Set of sample vertices/edges with VIE sampling
$\mathbf{t}_v\,/\mathbf{t}_e$	Probability distribution vector of triangle counts per vertex/edge
s	The superscript s denote the analogous sample quantity
^	The symbol ^ represents the estimator of the quantity

asymptotic estimation methods have been derived for the three sampling designs. Estimation from multiple samples is also presented here for the first time. The sampling algorithms to implement VIE sampling in the several network access scenarios are presented according to the different designs and estimation methods. Finally, we added several numerical results and provided more details and explanation throughout the text.

2. Quantities of interest and sampling

For ease of reading, Table 1 lists the main mathematical notation used throughout the paper. The rest of this section provides their definitions.

2.1 Triangle counts per vertex and edge

We represent a network under study as a graph $G = (V, E)$, where V is the set of vertices (nodes) and E is the set of edges. The graph is assumed to be unweighted, undirected, and without loops or multiple edges (simple). Let $N = |V|$ and $M = |E|$ denote the numbers of vertices and edges, respectively, which are assumed to be known for simplicity. The numbers of triangles of a vertex $w \in V$ and an edge $(u, v) \in E$ are defined by

$$T_w = |\{(u, v) \in E : (u, w), (v, w) \in E\}|, \quad w \in V \tag{1}$$

$$T_{(u,v)} = |\{w \in V : (u, w), (v, w) \in E\}|, \quad (u, v) \in E \tag{2}$$

Let

$$T_v(j) = \sum_{w \in V} 1_{\{T_w = j\}}, \quad j = 0, \ldots, N_t \tag{3}$$

denote the total number of vertices with j triangles, referred to as triangle count per vertex, where $N_t = \max\{T_w\}$ is the maximum number of triangles of a vertex in G. The proportion of vertices participating in j triangles is

$$t_v(j) = \frac{T_v(j)}{\sum_{i=1}^{N_t} T_v(i)} = \frac{1}{N} \sum_{w \in V} 1_{\{T_w = j\}}, \quad j = 0, \ldots, N_t \tag{4}$$

Similarly, let

$$T_e(j) = \sum_{(u,v) \in E} 1_{\{T_{(u,v)} = j\}}, \quad j = 0, \ldots, M_t \tag{5}$$

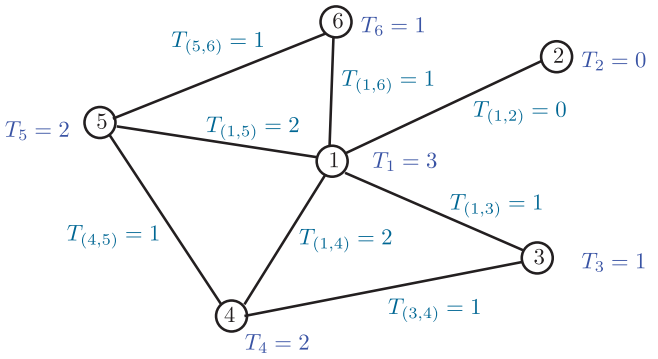

Figure 1. Illustration of the number of triangles for each vertex and edge in a small network. The triangle counts per vertex are $T_v(0) = 1$, $T_v(1) = 2$, $T_v(2) = 2$, $T_v(3) = 1$, and per edge are $T_e(0) = 1$, $T_e(1) = 5$, and $T_e(2) = 2$. The values of t_v (t_e, respectively) are obtained by dividing those of T_v (T_e, respectively) by N (M, respectively).

denote the total number of edges with j triangles, referred to as triangle count per edge, where $M_t = \max\{T_{(u,v)}\}$ is the maximum number of triangles per edge. The proportion of edges participating in j triangles is

$$t_e(j) = \frac{T_e(j)}{\sum_{j=1}^{M_t} T_e(j)} = \frac{1}{M} \sum_{(u,v)\in E} 1_{\{T_{(u,v)}=j\}}, \quad j = 0, \ldots, M_t \tag{6}$$

See Figure 1 for an illustration of the quantities defined.

We are interested in the *distribution of triangle counts per vertex*

$$\mathbf{t}_v = (t_v(0), \ldots, t_v(N_t))'$$

and the *distribution of triangle counts per edge*

$$\mathbf{t}_e = (t_e(0), \ldots, t_e(M_t))'$$

In principle, the largest possible value for M_t in a simple network is $N - 2$ and we assume that in practice, N_t and M_t may be known, or set large enough.

2.2 Sampling framework

Sampling has been used extensively for large complex networks in order to produce network statistics from a portion of the network which, if computed for the full graph, would be prohibitively expensive. When the only source of randomness in the network is the sampling design, the estimation procedures constructed under this assumption are refereed to as *design-based procedures*. In order to incorporate other sources of randomness into the network, such as the topology of the graph G, one generally specifies a model, leading to the so-called *model-based procedures*. We focus on design-based procedures in this work.

We introduce and focus on a general sampling framework for estimating the distributions of triangle counts of interest. The basic idea follows the work of Buriol et al. (2006), where to estimate the total number of triangles in the network, pairs consisting of a vertex and an edge are sampled to check if they form a triangle. We suppose that there are two separate sets of units being sampled, vertices and edges. Our sampling method can be characterized as having two stages: a selection stage, followed by an observation stage. More precisely, a sample of vertices is selected from V, which yields the set of *sampled vertices* V^*, and a sample of edges is selected from E, resulting in the set of *sampled edges* E^*. The manner in which vertices and edges are sampled can differ. Then, for each sampled vertex $w \in V^*$, we count the number of triangles of w formed with the sampled edges (u, v) in E^*, by checking if the edges (w, u) and (w, v) exist in the graph. Similarly, for each sampled edge $(u, v) \in E^*$, we count the number of triangles of (u, v) formed with the sampled vertices w in V^*, by observing if the edges (w, u) and (w, v) belong to E. We call this sampling method the VIE sampling.

We shall consider three main sampling designs of the graph G to obtain V^* and E^*. In the first design, n vertices and m edges are sampled from V and E uniformly at random, but *without replacement*. This situation arises naturally in static networks with full access and ensures that all units in V^* and E^* are distinct. In other contexts, while units are sampled with equal probability, sampling is done *with replacement*. For example, in static networks with restricted access, random walks are used to crawl the network. These random walks share important proprieties with random sampling with replacement. The standard random walk on vertices (selecting any of their neighbors with equal probabilities) samples (visits) edges uniformly at random with replacement in its stationary regime. By employing the standard Metropolis–Hastings (MH) technique to correct the bias in the standard random walk, vertices can also be sampled (visited) uniformly at random with replacement in the stationary regime. In *Bernoulli sampling*, each unit, vertex or edge, of the graph is subjected to an independent Bernoulli trial, with probabilities p_v or p_e, respectively, which determines whether the unit becomes part of the sample sets V^* and E^*. Sampling conducted in this manner appears in the case of streaming graphs. A random sample size is a potential drawback of Bernoulli sampling; however, this is counterbalanced by the ease of Bernoulli sampling and the benefits of guaranteed bounds on the sample size. Finally, we will also consider a *mixed sampling* of Bernoulli and with replacement samplings motivated by a random walk algorithm proposed in Section 5. While the considered sampling designs are relatively simple, they are by no means uncommon, and occur under a variety of scenarios described above.

For any of the considered sampling designs, the measured sample analogues of Equations (1) and (2) are

$$T_w^s = \begin{cases} |\{(u,v) \in E^* : (u,w),(v,w) \in E\}|, & w \in V^*, \\ 0, & w \notin V^* \end{cases} \tag{7}$$

and

$$T_{(u,v)}^s = \begin{cases} |\{w \in V^* : (u,w),(v,w) \in E\}|, & (u,v) \in E^*, \\ 0, & (u,v) \notin E^* \end{cases} \tag{8}$$

Note that the quantities (7) and (8) are not those in Equations (1) and (2) defined for the graph resulting from the VIE sampled edges and vertices: for example, a triangle formed just by the edges of E^* (with no vertices in V^*) would not be counted in either Equation (7) or Equation (8).

The sample triangle count per vertex is

$$T_v^s(i) = \sum_{w \in V} 1_{\{T_w^s = i\}}, \quad i = 0, \ldots, N_t^s \tag{9}$$

where N_t^s need not be the same as N_t and could even be larger as in sampling with replacement. Similarly to Equation (4), let

$$t_v^s(i) = \frac{T_v^s(i)}{\sum_{j=1}^{N_t^s} T_v^s(j)} = \frac{1}{N} \sum_{w \in V} 1_{\{T_w^s = i\}}, \quad i = 0, \ldots, N_t^s \tag{10}$$

be the proportion of vertices participating in i triangles after sampling and set $\mathbf{t}_v^s = (t_v^s(0), \ldots, t_v^s(N_t^s))'$.

The sample triangle count per edge is defined by

$$T_e^s(i) = \sum_{(u,v) \in E} 1_{\{T_{\{(u,v)\}}^s = i\}}, \quad i = 0, \ldots, M_t^s \tag{11}$$

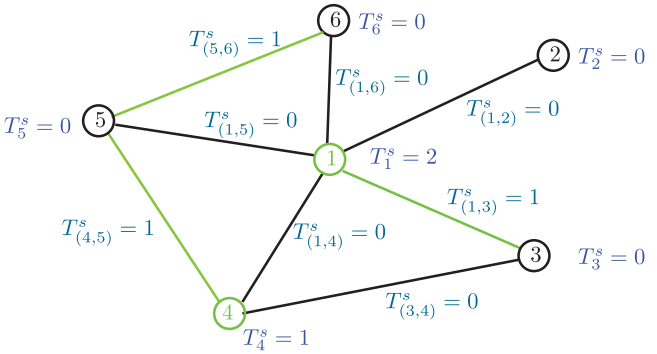

Figure 2. Illustration of a sample with $n = 2$ vertices (in green) and $m = 3$ edges (in green) selected at random from V and E, without replacement, yielding the sets $V^* = \{1, 4\}$ and $E^* = \{(1, 3), (4, 5), (5, 6)\}$. The sample triangle counts per vertex are $T^s_v(0) = 4$, $T^s_v(1) = 1$, $T^s_v(2) = 1$, and per edge $T^s_e(0) = 5$, $T^s_e(1) = 3$. The values of t^s_v (t^s_e, respectively) are obtained by dividing those of T^s_v (T^s_e, respectively) by N (M, respectively).

where M^s_t depends on the sampling design. Finally, let

$$t^s_e(i) = \frac{T^s_e(i)}{\sum_{j=1}^{M^s_t} T^s_e(j)} = \frac{1}{M} \sum_{(u,v)\in E} 1_{\{T^s_{(u,v)}=i\}} \tag{12}$$

denote the proportion of edges participating in i triangles after sampling and set $\mathbf{t}^s_e = (t^s_e(0), \ldots, t^s_e(M^s_t))'$. See Figure 2 for an illustration of the quantities defined. Our goal is to estimate $\mathbf{t}_v$ and $\mathbf{t}_e$ defined in Section 2.1 from the observed sample quantities (7)–(12).

2.3 Comparison with other sampling approaches

When sampling without replacement or with Bernoulli sampling, a subgraph $G^s = (V^s, E^s)$ can be constructed from the sampled vertices $w \in V^*$ and edges $(u, v) \in E^*$, along with the edges (w, u) and (w, v) if they belong both to E. We would like to contrast this subgraph with other classical approaches like induced or incident subgraph sampling (e.g. Kolaczyk, 2009, Chapter 5.3.1).

For example, with Bernoulli sampling when $p_v = 0$ and $p_e > 0$, observe that only edges are sampled and that G^s is the so-called incident subgraph sampling. But we note again that the quantities T^s_w and $T^s_{(u,v)}$ of interest are not triangle counts in the resulting subgraph (since no vertices are sampled when $p_v = 0$, $T^s_w = 0$ and $T^s_{(u,v)} = 0$ for all vertices w and edges (u, v) in G^s). In fact, for incident or induced subgraph sampling, if we considered the triangle counts per vertex (in the resulting subgraph), this setting would not be amenable to theoretical analysis, in the sense that we could not relate the count distributions per vertex in the subgraph to those in the original graph (as in Section 3 for the VIE sampling).

We comment here further on our choice of the sampling schemes. For example, for estimating $\mathbf{t}_v$, a natural and simple sampling scheme could consist of sampling vertices and then for each sampled vertex, calculating the exact number of triangles (by checking all pairs of its neighbors and seeing how many of these also connect) and then just using the empirical distribution of these as an estimate of $\mathbf{t}_v$. This sampling scheme, however, may not be practical, especially for large networks with heavy-tailed degree distributions, since checking all pairs of neighbors becomes prohibitively expensive for vertices with large degrees. Furthermore, this scheme would not work for sampling a streaming graph in one pass which is also an aim of this work. In contrast, with VIE sampling, only the sampled edges in E^* with incident vertices neighboring a sampled vertex, are used to count the number of triangles, mitigating the aforementioned computational issue. On the other hand, for estimating $\mathbf{t}_e$, note that sampling an edge and then calculating the exact number of triangles is not as big of an issue, particularly in networks showing disassortative mixing, since for each sampled edge, it is less likely that both of its vertices have large degrees. We will see that this case can be emulated with VIE sampling (see Section 3.1.2).

Another possibility, as noted in Section 1, would be to estimate triangle counts for sampled vertices and then translating these to triangle count distributions. This is the basis for the method

proposed in Section 4 below, using VIE sampling with multiples samples, which also serves as a baseline that other estimators are compared to (Section 3). We compared our approach with the work (Lim et al., 2018, algorithm Mascot-C, p. 7), where edges are sampled with constant probability and the number of local triangles for each vertex are estimated from the sampled graph. However, the estimation of triangle count distribution per vertex was worse than for VIE sampling using multiple samples (and, in fact, even when using a single sample). This is due in part to the estimation of the local triangles for all vertices in the sampled graph, while in our case, we fixed the sampled vertices and repeat the sampling of the edges reducing the variability (see Section 4).

3. Estimation from a single sample

It seems natural to estimate simply $t_v(j)$ by a "plug-in" value $t_v^s(j)$. This estimation, however, is biased. This line of reasoning is the focus of research on representative sampling (see e.g. Leskovec & Faloutsos, 2006), that studies how the topological properties of samples differ from those of the original network. Unfortunately, in estimation of the characteristics of our interest, many properties of the network are not captured from samples (unless standard assumptions of samples with i.i.d. observations are possible).

One possible estimation strategy can be based on appropriate adjustments of the "plug-in" estimators to correct the bias. Deriving such corrections for network total counts (e.g. total number of triangles) is often relatively straightforward but this is more challenging for distributions. Furthermore, the performance of the correction depends on the topology of the network, the distribution of interest, the sampling design, and their interactions. In this section, we propose two estimation approaches to correct the bias of the observed quantities using one sample from VIE sampling. The first approach is based on the inversion of an exact relation and the second approach on the asymptotic equivalence between the (expected) sample and true quantities.

3.1 Inversion estimation

In this section, we present an approach to estimate $\mathbf{t}_v$ and $\mathbf{t}_e$ from sample distributions $\mathbf{t}_v^s$ and $\mathbf{t}_e^s$. The approach is based on inversion and is common in similar problems (e.g. Antunes & Pipiras, 2016; Tune & Veitch, 2011; Zhang et al., 2015).

3.1.1 Distribution of triangle counts per vertex

Under the considered sampling designs, let $P_v(i, j)$ be the probability that a vertex with j triangles in G is observed to have i sampled triangles. If w^* is a vertex selected at random after a VIE sampling realization, then conditioning on its number of triangles in G leads to the relation

$$\mathbb{P}(T_{w^*}^s = i) = \sum_{j=0}^{N_t} P_v(i, j) t_v(j), \qquad i = 0, \ldots, N_t^s \tag{13}$$

Note also that

$$\mathbb{P}(T_{w^*}^s = i) = \mathbb{E}1_{\{T_{w^*}^s = i\}} = \mathbb{E}\frac{1}{N}\sum_{w \in V} 1_{\{T_w^s = i\}} = \mathbb{E}(t_v^s(i)) \tag{14}$$

where the last expectation $\mathbb{E}$ is with respect to the randomness in the sampling. The relations (13)–(14) can be combined into a matrix form as

$$\mathbb{E}(\mathbf{t}_v^s) = \mathbf{P}_v \mathbf{t}_v \tag{15}$$

where $\mathbf{P}_v = (P_v(i,j))$ is a $(N_t^s + 1) \times (N_t + 1)$ matrix specific to the sampling design defined below. This suggests that an estimator of the triangle count distribution per vertex can be defined as

$$\hat{\mathbf{t}}_v = \mathbf{P}_v^+ \mathbf{t}_v^s \tag{16}$$

where $\mathbf{P}_v^+ = (\mathbf{P}_v' \mathbf{P}_v)^{-1} \mathbf{P}_v'$ is the left generalized inverse of $\mathbf{P}_v$. The estimator (16) is unbiased, that is, $\mathbb{E}(\hat{\mathbf{t}}_v) = \mathbf{P}_v^+ \mathbb{E}\mathbf{t}_v^s = \mathbf{P}_v^+ \mathbf{P}_v \mathbf{t}_v = \mathbf{t}_v$ by Equation (15) and its variance(-covariance) matrix is

$$\begin{aligned}
\mathbb{V}(\hat{\mathbf{t}}_v) &= \mathbb{E}((\hat{\mathbf{t}}_v - \mathbb{E}(\hat{\mathbf{t}}_v))(\hat{\mathbf{t}}_v - \mathbb{E}(\hat{\mathbf{t}}_v))') = \mathbb{E}[(\mathbf{P}_v^+ \mathbf{t}_v^s - \mathbf{t}_v)(\mathbf{P}_v^+ \mathbf{t}_v^s - \mathbf{t}_v)'] \\
&= \mathbf{P}_v^+ \mathbb{E}(\mathbf{t}_v^s (\mathbf{t}_v^s)')(\mathbf{P}_v^+)' - \mathbf{t}_v (\mathbf{t}_v)' = \mathbf{P}_v^+ (\mathbb{V}(\mathbf{t}_v^s) + \mathbb{E}(\mathbf{t}_v^s)\mathbb{E}(\mathbf{t}_v^s)')(\mathbf{P}_v^+)' - \mathbf{t}_v (\mathbf{t}_v)' \\
&= \mathbf{P}_v^+ \mathbb{V}(\mathbf{t}_v^s)(\mathbf{P}_v^+)'
\end{aligned} \tag{17}$$

Sampling without replacement

If n vertices and m edges are sampled uniformly, without replacement, then $\mathbf{P}_v$ is a $(N_t^s + 1) \times (N_t + 1)$ matrix, with $N_t^s = \min(N_t, m)$ and entries

$$P_v(i,j) = \begin{cases} \left(1 - \frac{n}{N}\right) 1_{\{i=0\}} + \frac{n}{N} \frac{\binom{j}{i}\binom{M-j}{m-i}}{\binom{M}{m}}, & j = 0, \ldots, N_t,\ i = 0, \ldots, j, \\ 0, & \text{otherwise}, \end{cases} \tag{18}$$

with the convention that $\binom{a}{b}$ equals to 0 when either $a < 0$ or $a > b$. The representation (18) can be explained as follows. Since there are $\binom{N}{n}$ possible samples of size n and $\binom{N-1}{n-1}$ samples may be chosen to include a given vertex, it follows that the probability of a vertex being sampled is n/N. Note that the term $\binom{j}{i}\binom{M-j}{m-i}/\binom{M}{m}$ corresponds to the probability of sampling i triangles out of j (or, equivalently, sampling i edges out of j that form triangles with a vertex in question), which is given by the hypergeometric distribution with parameters j, m, and M. Similarly, $1 - n/N$ is the probability of not sampling a vertex, for which the number of sample triangles will be 0 with probability 1 and hence the term $1_{\{i=0\}}$.

Sampling with replacement

In the case when n vertices and m edges are sampled with replacement, the matrix $\mathbf{P}_v$ has dimension $(m+1) \times (N_t + 1)$, with entries

$$P_v(i,j) = \begin{cases} \left(1 - \frac{1}{M}\right)^n 1_{\{i=0\}} + \left(1 - \left(1 - \frac{1}{M}\right)^n\right)\binom{m}{i}\left(\frac{j}{M}\right)^i \left(1 - \frac{j}{M}\right)^{m-i}, & j = 0, \ldots, N_t, \\ & i = 0, \ldots, m, \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

Indeed, arguing as for Equation (18), the probability of sampling a vertex is $1 - \left(1 - \frac{1}{M}\right)^n$. The part containing this term in Equation (19) also includes the probability of sampling i triangles which is given by the binomial distribution with parameters m and j/M.

Bernoulli sampling

If each vertex and edge are sampled with probabilities p_v and p_e, respectively, then $\mathbf{P}_v$ is a square $(N_t + 1) \times (N_t + 1)$ matrix, with entries

$$P_v(i,j) = \begin{cases} (1 - p_v) 1_{\{i=0\}} + p_v \binom{j}{i} p_e^i (1 - p_e)^{j-i}, & j = 0, \ldots, N_t,\ i = 0, \ldots, j, \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

which follows by similar arguments as for the other designs.

3.1.2 Distribution of triangle counts per edge

For any of the considered sampling designs, if $(u, v)^*$ is an edge selected at random after a VIE sampling realization, the same arguments as for Equations (13)–(14) lead to

$$\mathbb{P}(T^s_{(u,v)^*} = i) = \mathbb{E}(t^s_e(i)) = \sum_{j=0}^{M_t} P_e(i, j)t_e(j), \qquad i = 0, \ldots, M^s_t \tag{21}$$

where $P_e(i, j)$ is the probability that an edge with j triangles participates in i sampled triangles. Then, the relation

$$\mathbb{E}(\mathbf{t}^s_e) = \mathbf{P}_e\mathbf{t}_e \tag{22}$$

holds, similarly to Equation (15), where $\mathbf{P}_e = (P_e(i, j))$ is a matrix with dimensions $(M^s_t + 1) \times (M_t + 1)$ whose entries depend on the sampling design. The estimator of $\mathbf{t}_e$ is obtained simply by replacing the expected value by the observed value in Equation (22) and inverting, that is,

$$\hat{\mathbf{t}}_e = \mathbf{P}^+_e\mathbf{t}^s_e \tag{23}$$

where $\mathbf{P}^+_e = (\mathbf{P}^{'}_e\mathbf{P}_e)^{-1}\mathbf{P}^{'}_e$. Following the same reasoning around (16)–(17), the estimator in Equation (23) is unbiased with variance matrix,

$$\mathbb{V}(\hat{\mathbf{t}}_e) = \mathbf{P}^+_e\mathbb{V}(\mathbf{t}^s_e)(\mathbf{P}^+_e)^{'} \tag{24}$$

Sampling without replacement

In this design, $\mathbf{P}_e$ is a $(\min(M_t, n) + 1) \times (M_t + 1)$ matrix that can be obtained from Equation (18) by replacing N, n, M, m by M, m, N, n, respectively. This form of $\mathbf{P}_e$ can be argued in the same way as Equation (18) for $\mathbf{P}_v$.

Sampling with replacement

The matrix $\mathbf{P}_e$ has dimensions $(n + 1) \times (M_t + 1)$ and the form (19) by replacing N, n, M, m by M, m, N, n, respectively.

Bernoulli sampling

The matrix $\mathbf{P}_e$ is $(M_t + 1) \times (M_t + 1)$ and has the form (20) by replacing p_v, p_e by p_e, p_v, respectively.

Mixed sampling

We will propose an algorithm in Section 5, which emulates the case of Bernoulli sampling of vertices with $p_v = 1$ and edges with replacement. The corresponding matrix $\mathbf{P}_e$ is $(M_t + 1) \times (M_t + 1)$ and has nonzero entries

$$P_e(i, i) = \left(1 - \frac{m}{M}\right) 1_{\{i=0\}} + \frac{m}{M}, \quad P_e(0, j) = 1 - \frac{m}{M},\ j \geq 1 \tag{25}$$

This could be seen from Equation (20) with p_v and p_e replaced by $p_e = m/M$ and $p_v = 1$, respectively, or just by arguing directly. It can also be checked that $\mathbf{P}^{-1}_e$ has a closed form with the nonzero entries

$$P^{-1}_e(0, 0) = 1, \quad P^{-1}_e(0, j) = 1 - \frac{M}{m}, \quad P^{-1}_e(i, i) = \frac{M}{m} \tag{26}$$

In this case, since only $P^{-1}_e(i, i)$ is nonzero for the ith row of $\mathbf{P}^{-1}_e$ $(i \neq 0)$, the resulting estimator (23) (except $\hat{t}_e(0)$) is just a (scaled) empirical distribution of the true triangle counts per sampled edges.

Remark 1. Our distinction between samplings with and without replacement might appear somewhat restrictive in the following sense. Note that a sampling procedure with replacement can be carried out (e.g. for distribution of triangle counts per vertex) but then only distinct sampled edges be kept for inference under "sampling without replacement," with the latter referring rather to how collected information is used. In fact, one could develop an analogous inversion approach for such sampling schemes as well, with their own probability matrices $\mathbf{P}_v$ and $\mathbf{P}_e$. But we found the "sampling without replacement" inference for these sampling schemes to be slightly inferior to that with replacement when all sampled information collected without replacement is used. This probably should not be too surprising but might also go against some of the current practices (e.g. Zhang et al., 2015) where inference is made on sample subgraph even if vertices/edges repeat in sampling with replacement as when using a random walk. For the above reason and for simplicity sake, we decided not to include sampling schemes with replacement where only distinct sampled units are kept for inference.

3.1.3 Regularization approach

The performance of the inversion estimators in Equations (16) and (23) can be viewed through the matrix $\mathbf{P}_v$ and $\mathbf{P}_e$ condition numbers (i.e. the ratio of the largest to smallest singular values). For instance, the numerical inversion of $\mathbf{P}_v^{'}\mathbf{P}_v$ in Equations (16) and (17) is governed by its condition number, which is the square of the condition number of $\mathbf{P}_v$. If the condition number is small, the matrix is well-conditioned and its inverse can be computed accurately. If the condition number is large, then the matrix is said to be ill-conditioned. The matrices $\mathbf{P}_v$ and $\mathbf{P}_e$ are ill-conditioned for smaller values of n and m or p_v and p_e. This means that computation of the inverse of $\mathbf{P}_v^{'}\mathbf{P}_v$ is prone to large numerical errors that will produce estimates (16) with an oscillating behavior. Regularization is a common method used to solve ill-posed problems. Since the use of this technique is analogous for both cases, we consider only the triangle count distribution per vertex.

We solve our linear inverse problem from a penalized weighted least squares perspective with nonnegative constraints, where a regularization (penalty) term is added to the objective function and a penalization parameter controls the degree of the regularization. More specifically, the penalized estimator $\hat{\mathbf{t}}_v$ is defined as the solution of the optimization problem

$$\underset{\mathbf{t}}{\operatorname{argmin}}\ (\mathbf{t}_v^s - \mathbf{P}_v\mathbf{t})'\mathbf{W}^{-1}(\mathbf{t}_v^s - \mathbf{P}_v\mathbf{t}) + \lambda\phi(\mathbf{t}) \tag{27}$$

subject to $t(i) \geq 0,\ i = 0, 1, \ldots, N_t,\ \sum_{i=0}^{N_t} t(i) = 1$, where $\mathbf{W}$ is a matrix representing suitable weights taken here to be a diagonal matrix with entries $\mathbf{t}_v^s$, $\phi(\mathbf{t})$ refers to the function regularizing $\mathbf{t}$, and $\lambda > 0$ is a scalar penalty, that sets the degree of regularization, to be determined separately. When $\lambda = 0$, $\mathbf{W} = \mathbf{I}$ and the constraints are ignored, the minimization (27) yields the inversion estimator (16). For the inversion problems considered in this work, however, regularization with $\lambda > 0$ is critical; the inversion estimator with $\lambda = 0$ is usually impractical due to a large variance. The regularization with $\lambda > 0$ reduces the variance but also introduces some bias, with λ chosen to balance the variance–bias trade-off. The triangle count distributions encountered in practice are usually smooth (i.e. $t_v(i) \approx t_v(j)$ for i and j close)—see Section 6. A convenient regularization is the use of the quadratic function

$$\phi_{\text{quad}}(\mathbf{t}) = \sum_{i=0}^{N_t-1} (t(i+1) - t(i))^2 \tag{28}$$

that forces estimates to be smooth and significantly reduces their variance. The optimization problem (27)–(28) can be written as a quadratic program and solved for $\mathbf{t}$ using standard software. For the implementation details, we refer the reader to Zhang et al. (2015), where regularization is

used in a similar context to estimate the degree distribution when sampling vertices and edges. The parameter λ is chosen based on Stein's unbiased risk estimation (SURE) method proposed by Eldar (2009). The use of this method is explained in Section 3.2 of Zhang et al. (2015). The choice of $\mathbf{t}_v^s$ for the diagonal of $\mathbf{W}$ also follows Zhang et al. (2015).

3.2 Asymptotic estimation

The regularization approach tends to perform poorly at the distribution tail (especially when the latter is heavy-tailed). Estimation in the tail is addressed in this section. We relate the tails of the distributions of triangle counts per vertex and edge with the respective tails of sample distributions. Since the latter is observable, the relation can be used to estimate the former. We also consider the case when the original distribution has a power-law tail.

3.2.1 Distribution of triangle counts per vertex

The sample triangle count of a sampled vertex w^* chosen at random from V^* after a VIE sampling realization, can be expressed as

$$T_{w^*}^s = \sum_{(u,v)\in E^*} 1_{\{(u,w^*),(v,w^*)\in E\}} = \sum_{(u,v),(u,w^*),(v,w^*)\in E} 1_{\{(u,v)\in E^*\}}, \qquad w^* \in V^* \tag{29}$$

Note that $\mathbb{E}1_{\{(u,v)\in E^*\}} = \pi_e$, where π_e is the probability of sampling an edge under a given sampling design. Furthermore, since the second sum in Equation (29) is over the triangles including the vertex w^*, it has T_{w^*} terms. It follows that

$$T_{w^*}^s = \pi_e T_{w^*} + \sum_{(u,v),(u,w^*),(v,w^*)\in E} (1_{\{(u,v)\in E^*\}} - \pi_e), \qquad w^* \in V^* \tag{30}$$

Since the second term in Equation (30) can be thought as approximately Gaussian with standard deviation of the order $\sqrt{T_{w^*}} \ll T_{w^*}$ for large T_{w^*}, we expect that

$$\mathbb{P}(T_{w^*} \le i) \sim \mathbb{P}(T_{w^*}^s \le i\pi_e), \qquad w^* \in V^* \tag{31}$$

for large i. Viewing Equation (31) as a relation for continuous random variables, by differentiation we have

$$\mathbb{P}(T_{w^*} = i) \sim \pi_e \mathbb{P}(T_{w^*}^s = i\pi_e), \qquad w \in V^* \tag{32}$$

Since w^* is a sampled vertex, $\mathbb{P}(T_{w^*} = i) = t_v(i)\pi_v$, where π_v is the probability that a vertex is included in the sample for a given sampling design. On the other hand, by Equation (14), $\mathbb{P}(T_{w^*}^s = i\pi_e) = \mathbb{E}(t_v^s(i\pi_e))$. Then, we can write the asymptotic relation (32) as

$$t_v(i) \sim \pi_e \pi_v^{-1} \mathbb{E}(t_v^s(i\pi_e)) \tag{33}$$

for large i. Therefore, for large i,

$$\hat{t}_v(i) = \pi_e \pi_v^{-1} t_v^s(i\pi_e) \tag{34}$$

is a natural estimator of the original distribution tail obtained through the scaling of the empirical sample distribution $\mathbf{t}_v^s$.

If the distribution of triangle counts per vertex has a power-law tail with parameter β, that is,

$$t_v(i) \sim c\beta i^{-\beta-1} \tag{35}$$

for large i, where $\beta > 0$ and $c > 0$, then Equation (33) implies

$$\mathbb{E}(t_v^s(i)) \sim \pi_v^{-1} \pi_e^{-\beta} c\beta i^{-\beta-1} \tag{36}$$

This means that the sample distribution $\mathbf{t}_v^s$ is expected to have a power-law tail as well with the same parameter β, which can be estimated directly from the empirical sample distribution.

Sampling designs

When random sets of n vertices and m edges are selected from V and E with or without replacement, $\pi_v = n/N$ and $\pi_e = m/M$. With Bernoulli sampling, we have $\pi_v = p_v$ and $\pi_e = p_e$.

3.2.2 Distribution of triangle counts per edge

The sample triangle count of a sampled vertex $(u, v)^*$ chosen at random from E^* after a VIE sampling realization, is given by

$$T^s_{(u,v)^*} = \sum_{v \in V^*} 1_{\{(u,w),(v,w) \in E\}}, \quad (u, v)^* \in E^* \tag{37}$$

Following similar arguments as for Equations (30)–(32), we get

$$\mathbb{P}(T_{(u,v)^*} = i) \sim \pi_v \mathbb{P}(T^s_{(u,v)^*} = \pi_v i), \qquad (u, v)^* \in E^* \tag{38}$$

for large i. Therefore, we have the relation

$$t_e(i) \sim \pi_v \pi_e^{-1} \mathbb{E}(t^s_e(\pi_v i)) \tag{39}$$

and an estimator for the distribution tail

$$\hat{t}_e(i) = \pi_v \pi_e^{-1} t^s_e(\pi_v i) \tag{40}$$

for large i, through the scaling of the empirical distribution of $t^s_e(i)$. The probabilities π_v, π_e depend on the sampling design as discussed at the end of Section 3.2.1.

Additionally, if

$$t_e(i) \sim c\gamma i^{-\gamma-1} \tag{41}$$

for large i, where $\gamma > 0$ and $c > 0$, then

$$\mathbb{E}(t^s_e(i)) \sim \pi_e^{-1} \pi_v^{-\gamma} cC\gamma t^{-\gamma-1} \tag{42}$$

which also has the same power-law exponent.

3.2.3 Relations between power-law exponents

The reference to and relevance of power-law tails above should not surprise the reader. On one hand, examples of such real networks appear in Section 6 below. On the other hand, such tails are also expected for the following reason. It is well known (e.g. Newman, 2018, Chapter 10) that the degree distributions of real networks can have power-law tails. That is, if D_{w^*} denotes the degree of a randomly selected vertex w^*, then

$$\mathbb{P}(D_{w^*} = k) \sim c_0 \alpha k^{-\alpha-1} \tag{43}$$

for large k, where $c_0 > 0$, $\alpha > 1$. Furthermore, one commonly finds the clustering coefficient of a vertex w^*, that is, $T_{w^*}/\binom{D_{w^*}}{2}$ or $T_{w^*}/D^2_{w^*}$ to be roughly ξ_{w^*} where ξ_{w^*} varies over a limited range. Then, $T_{w^*} \sim \xi_{w^*} D^2_{w^*}$ and, by conditioning on ξ_{w^*} and using Equation (43),

$$t_v(i) \sim \mathbb{P}\left(D_{w^*} = \frac{i^{1/2}}{\xi^{1/2}} \right) \sim c_0 \alpha (\mathbb{E}\xi_v^{-\alpha/2-1}) i^{-\alpha/2-1} \tag{44}$$

for large i. In particular, note that the tail exponent β of the distribution of triangle counts per vertex relates to α as

$$\beta = \alpha/2. \tag{45}$$

This is also what we typically observe for real networks. Relationship between the tail exponent γ of the distribution of triangle counts per edge and the tail exponent α appears to be more delicate, and will not be discussed here.

4. Estimation from multiple samples

In this section, we study the feasibility of a single approach to estimate the bulk and the tail of the distribution of triangle counts per vertex and edge, at the cost of using several VIE samples collected in parallel. The main idea is to correct the bias of the sample quantities T_w^s and $T_{(u,v)}^s$ through the use of standard weighted averaging techniques, and then form the empirical distributions using the scaled numbers of triangles of the vertices and edges sampled. Since this can be applied to a more general sampling scenario, we consider that each unit $w \in V$ and $(u, v) \in E$ can have unequal probability of being included in the sample.

4.1 Distribution of triangle counts per vertex

We shall first define estimators of interest for a single sample and then consider their averages from multiple samples. Suppose that, under a given sampling design (without replacement), each edge $(u, v) \in E$ has probability $\pi_e(u, v)$ of being included in the sample E^*. Then, the Horvitz–Thompson (HT) estimator (see e.g. Thompson, 2012; Tillé, 2006) of the total number of triangles of a sampled vertex w takes the form

$$\widehat{T}_w = \sum_{(u,v)\in E^*} \frac{1_{\{(u,w),(v,w)\in E\}}}{\pi_e(u,v)} = \sum_{(u,v),(u,w),(v,w)\in E} \frac{1_{\{(u,v)\in E^*\}}}{\pi_e(u,v)}, \qquad w \in V^* \tag{46}$$

The estimator (46) is an unbiased estimator of T_w, since $\mathbb{E}1_{\{(u,v)\in E^*\}} = \pi_e(u, v)$ and hence

$$\mathbb{E}(\widehat{T}_w) = \mathbb{E}\left(\sum_{(u,v),(u,w),(v,w)\in E} \frac{1_{\{(u,v)\in E^*\}}}{\pi_e(u,v)} \right) = T_w \tag{47}$$

If $\pi_e((u, v), (u', v'))$ denotes the probability that edges (u, v) and (u', v') are both in the sample E^*, then the variance of the HT estimator $\widehat{T}_w$ can be expressed as

$$\mathbb{V}(\widehat{T}_w) = \sum_{(u,v),(u,w),(v,w)\in E} \sum_{(u',v'),(u',w),(v',w)\in E} \frac{\pi_e((u,v),(u',v')) - \pi_e(u,v)\pi_e(u',v')}{\pi_e((u,v))\pi_e(u',v')} \tag{48}$$

with $\pi_e((u, v), (u', v')) = \pi_e(u, v)$ for convenience when $(u, v) = (u', v')$.

Sampling without replacement

In this sampling design, we have $\pi_e(u, v) = m/M$ and $\pi_e((u, v), (u', v')) = m(m - 1)/(M(M - 1))$. It can be shown that Equation (48) reduces to

$$\mathbb{V}(\widehat{T}_w) = \frac{M}{m}\left(1 - \frac{m}{M}\right) T_w \left(1 - \frac{T_w - 1}{M - 1}\right) \tag{49}$$

Bernoulli sampling

For this sampling design, $\pi_e(u, v) = p_e$ and $\pi_e((u, v), (u', v')) = p_e^2$. It follows from Equation (48) that

$$\mathbb{V}(\widehat{T}_w) = \frac{1 - p_e}{p_e} T_w \tag{50}$$

Designs with replacement

The HT estimator can also be applied to random sampling with replacement. In this case, the sum in Equation (46) is over the distinct edges in E^* and $\pi_e(u, v)$ is the probability of (u, v) being included in the sample E^*, which is equal to $1 - (1 - 1/M)^m$. Similarly, the probability $\pi_e((u, v), (u', v'))$ that both (u, v) and (u', v') are included in the sample E^* is given by $\pi_e(u, v) + \pi_e(u', v') - (1 - (1 - \pi_e(u, v) - \pi_e(u', v'))^m)$. However, in designs with replacement,

the so-called Hansen–Hurwitz (HH) estimation (see e.g. Thompson, 2012; Tillé, 2006) is often employed that includes all the sample collected. This is also in line with the analysis conducted in Section 3, where repetitions of sampling units are taken into account. The HH estimator for T_w is defined as

$$\widehat{T}_w = \frac{1}{m} \sum_{(u_i,v_i)\in E^*} \frac{1_{\{(u_i,w),(v_i,w)\in E\}}}{\pi_e(u_i,v_i)} = \frac{1}{m} \sum_{(u_i,v_i),(u_i,w),(v_i,w)\in E} \frac{1_{\{(u_i,v_i)\in E^*\}}}{\pi_e(u_i,v_i)}, \qquad w \in V^* \tag{51}$$

where $\pi_e(u_i, v_i)$ is the probability of edge (u_i, v_i) being sampled on the ith draw. Since $\mathbb{E}1_{\{(u_i,v_i)\in E^*\}} = \pi_e(u_i, v_i)$, the estimator is unbiased as in Equation (47) and its variance can be shown to be

$$\mathbb{V}(\widehat{T}_w) = \frac{1}{m} \sum_{(u,v)\in E} \pi_e(u,v) \left(\frac{1_{\{(u,w),(v,w)\in E\}}}{\pi_e(u,v)} - T_w \right)^2 \tag{52}$$

In the case of random sampling with replacement, we have $\pi_e(u, v) = 1/M$ which yields

$$\mathbb{V}(\widehat{T}_w) = \frac{T_w}{m}(M - T_w) \tag{53}$$

The comparison of the variances in Equations (49), (50), and (53) shows that sampling without replacement has the lowest variance for $m > 1$, followed by Bernoulli sampling if $T_w > m$ assuming $p_e = m/M$.

For a vertex $w \in V^*$, the variance of the estimator $\widehat{T}_w$ tends to increase with T_w. However, the performance of $\widehat{T}_w$ will also be poor for small values of T_w. The reason is twofold. First, for small probabilities of sampling edges or when the total number of triangles of a vertex is small, the observed triangles of a sampled vertex will likely be zero and thus so its estimate in Equation (46). On the other hand, when we scale the number of triangles observed by the probability of sampling an edge in Equation (46), there will be gaps between the possible values of the estimates. To improve estimation, we repeat the sampling of edges r times in parallel, increasing the sampling cost, and let

$$\widehat{T}^r_w = \frac{1}{r} \sum_{k=1}^{r} \widehat{T}^{(k)}_w \tag{54}$$

It is then natural to set

$$\widehat{T}_v(i) = \sum_{w\in V^*} \frac{1_{\{\lfloor \widehat{T}^r_w \rfloor = i\}}}{\pi_v(w)} \tag{55}$$

as an estimator for the count $T_v(i)$ of vertices with i triangles, where $\lfloor . \rfloor$ denotes the "floor" integer part and $\pi_v(w)$ are the vertex inclusion probabilities corresponding to the underlying network sampling design. These will be qual to n/N, for sampling with and without replacement, and p_v for Bernoulli sampling. The presence of $\pi_v(w)$ in Equation (55) is to compensate for the fact that the sum in Equation (55) is over $w \in V^*$ and hence only sampled vertices are included. The final probability estimator in this context takes the form

$$\widehat{t}_v(i) = \widehat{T}_v(i)/N \tag{56}$$

4.2 Distribution of triangle counts per edge

The analysis developed in the previous section holds for the distribution of triangle counts per edge. For instance, the HT estimator of the total number of triangles of a sampled edge (u, v) is

$$\widehat{T}_{(u,v)} = \sum_{w\in V^*} \frac{1_{\{(u,w),(v,w)\in E\}}}{\pi_v(w)}, \qquad (u,v) \in E^* \tag{57}$$

By averaging through r samples of vertices obtained in parallel, let $\widehat{T}^{r}_{(u,v)} = \frac{1}{r}\sum_{k=1}^{r}\widehat{T}^{(k)}_{(u,v)}$ and

$$\widehat{T}_e(i) = \sum_{(u,v)\in E^*} \frac{1_{\{\lfloor \widehat{T}^{r}_{(u,v)} \rfloor = i\}}}{\pi_e(u,v)} \tag{58}$$

from which the estimator is defined as $\widehat{t}_e(i) = \widehat{T}_e(i)/M$. Similar expressions for the HH estimator with a replacement design can be derived.

Remark 2. The HT or HH estimation developed above is quite general in its applicability. In fact, it is not limited to VIE sampling and applies when sampling probabilities $\pi_v(w)$ and $\pi_e(u,v)$ are different across w and (u,v). The latter occurs with other common network sampling designs, such as induced or incident subgraph sampling (see Kolaczyk, 2009, Chapter 5.3). However, we note again that these sampling methods are not amenable to the analysis developed in Section 3.

5. Algorithms

In this section, we show how to implement VIE sampling for several network access scenarios corresponding to the considered sampling designs. For restricted access and streaming graphs, the implementation is described through formal algorithms.

5.1 Case of a single sample

Sampling static graphs with full access

When any vertex or edge can be accessed directly in a static network, VIE sampling can be carried out through a simple algorithm that samples vertices and edges at random *without replacement*, and then for each sampled vertex in V^* counts the number of triangles formed with the sampled edges in E^* (and vice versa). Such full access to the network, however, may not be available in other scenarios, for example, when networks can only be crawled or when dealing with streaming edges.

Sampling static graphs with restricted access

Many real-world networks can only be crawled, i.e. one can only explore the neighbors of the visited vertices. In this context, sampling procedures are commonly based on random walks. It is assumed that access to one initial vertex is available and the network is connected or the largest giant connected component covers the majority of the network so that the disconnected parts can be ignored. Two independent random walks could be used to carry out VIE sampling with *replacement*: for instance, first performing a standard version of random walk sampling (i.e. selecting a vertex uniformly at random among the neighbors of the visited vertex) to sample edges uniformly at random (E^*); and then a random walk to sample vertices uniformly at random (V^*) using the MH algorithm. The number of triangles of each sampled vertex w formed with the sampled edges in E^* (and vice versa) can be incremented in each step of the MH algorithm by checking the neighborhood of w.

Alternatively, we propose here a scheme with a single random walk on edges, that implements *mixed* VIE sampling, that samples m edges at random with replacement and emulates $p_v = 1$—see Algorithm 1. Initially, an edge is selected at random and the sample sets V^*, E^* are empty (line 1). Then, for each sampled edge (u,v) in E^*, the number of common neighbors of u and v provides the number of triangles of that edge (line 5). Additionally, each common neighbor of u and v is added to V^* and the triangle that forms with the edge (u,v) is increased by one (lines 6–10). Finally, the current vertex is set to v and the next vertex is chosen at random among its neighbors

Algorithm 1: Mixed VIE sampling ($p_v = 1$, m) with random walk

Data: static graph G; **Result**: E^*, V^*, T^s_w and $T^s_{(u,v)}$

1 **Initialization:** $(u, v) \leftarrow rand(E)$, $V^* \leftarrow \emptyset$, $E^* \leftarrow \emptyset$;
2 **While** $|E^*| <= m$ **do**
3 $E^* \leftarrow E^* \cup \{(u, v)\}$;
4 **If** $(u, v) \notin E^*$ **then**
5 $T^s_{(u,v)} \leftarrow |Neighbor(u) \cap Neighbor(v)|$;
6 **ForEach** $k \in (Neighbor(u) \cap Neighbor(v))$ **do**
7 **If** $k \notin V^*$ **then**
8 $V^* \leftarrow V^* \cup \{k\}$;
9 $T^s_k \leftarrow 0$;
10 $T^s_k \leftarrow T^s_k + 1$;
11 $u \leftarrow v$;
12 $v \leftarrow rand(Neighbor(u))$;

representing the next sample edge (lines 11–12), until m edges have been sampled (line 2). After the algorithm is finished, the quantities T^s_w and $T^s_{(u,v)}$ are used to compute Equations (10) and (12), respectively, which then enter into the estimators developed in Sections 3 and 4.

We also analyze Algorithm 1 concerning its processing time per edge and total running time. The main time-intensive operation is to compute the set of common neighbors for two incident nodes of each sampled edge (u, v) (line 5). Let D_u denote the degree of node $u \in V$. Assuming that $D_u \leq D_v$, to compute the set $Neighbor(u) \cap Neighbor(v)$, each element of $Neighbor(u)$ is checked. If $Neighbor(v)$ contains the element, it is added to the set. Thus, the running time for processing a sampled edge is of the order $O(min(D_u, D_v)))$. The total running time of the algorithm is $O(D^*_{max} m)$, where $D^*_{max} = \max_{(u,v)\in E^*} (min(D_u, D_v))$. The algorithm requires memory space to store the sets $|V^*|$ and $|E^*| = m$ and the respective vertex and edge triangle counts. The value of $|V^*|$ depends on the network structure and cannot be computed explicitly.

Sampling streaming graphs

Many real-world networks naturally evolve over time, as new edges/vertices are added to the network. A natural representation of such networks (or streaming graphs) is in the form of a stream of edges. We shall describe how to perform VIE *Bernoulli* sampling to select a subgraph $G_s = (V^s, E^s)$ from G, which includes E^* and V^*, in one pass when G is presented as a stream of edges in no particular order. Two uniform random hash functions ($hash_v$ and $hash_e$) on $[0, 1]$ are used to sample vertices and edges at random with probabilities p_v and p_e, respectively, which are then added to the sample sets E^* and V^*—see Algorithm 2 (lines 3–4 and lines 5–6). Additionally, the sampled vertices are also added to V^s and edges to $G_s = (V^s, E^s)$ (lines 7–10). We note that in the streaming scenario if a vertex is sampled, its edges have to be added to the subgraph G^s (line 10). This is the cost of having a one pass algorithm over the input stream, in order to able to count the number of triangles of each sampled vertex in V^* formed with the sampled edges in E^* and vice versa (lines 11–12). The additional edges that do not enter into the VIE sampling can be deleted at the end of the stream but we omit this step in the algorithm. If two passes over the stream of edges are possible, these additional edges need not to be stored. The algorithm also applies to the case of a static graph with full access, as a one pass algorithm through the list of edges.

For Algorithm 2, the processing time per edge of the stream graph is not a time-consuming operation (lines 3–10). Another factor is the triangle count of vertices and edges (lines 11–12); however, compared with Algorithm 1, this is now done on the subgraph G^s with lower running time. The algorithm requires memory space to store G^s where the expectation of $|E^s|$ is given by

Table 2. Properties of the networks used in the experiments

Data set	Vertices (N)	Edges (M)	Triangles (total)
Synthetic network	20,000	350,000	643,711
Arxiv HEP-TH	8,638	24,806	27,869
gemsec-Facebook	50,515	819,090	2,273,700
com-Amazon	334,863	925,872	667,129

Algorithm 2: VIE sampling—Bernoulli (p_v, p_e) for streaming graphs

Data: streaming graph G; **Result**: T^s_w and $T^s_{(u,v)}$ from $G^s = (E^s, V^s)$, E^*, V^*.

Initialization: $V^s, E^s, V^*, E^* \leftarrow \emptyset$;
ForEach edge (u, v) from G **do**
  If $hash_v(u)$ (resp. $hash_v(v)$) $< p_v$ **and** u (resp. v) $\notin V^*$ **then**
    $V^* \leftarrow V^* \cup \{u\}$ (resp. $\{v\}$);
  If $hash_e(u, v) < p_e$ **then**
    $E^* \leftarrow E^* \cup \{(u, v)\}$;
  If $hash_v(u) < p_v$ **or** $hash_v(v) < p_v$ **or** $hash_e(u, v) < p_e$ **then**
    If $u \notin V^s$(resp. $v \notin V^s$) **then**
      $V^s \leftarrow V^s \cup \{u\}$(resp. $\{v\}$);
    $E^s \leftarrow E^s \cup \{(u, v)\}$;
$T^s_w \leftarrow |\{(u, v) \in E^*{:}(u, w), (v, w) \in E^s\}|, \ w \in V^*$
$T^s_{(u,v)} \leftarrow |\{w \in V^*{:}(u, w), (v, w) \in E^s\}|, \ (u, v) \in E^*$

M times the probability of adding an edge $(1 - (1 - p_e)(1 - p_v)^2)$ (the expectation of $|V^s|$ cannot be computed explicitly).

If we are interested to guarantee a bounded sample size of n vertices and m edges selected at random without replacement from a streaming graph, we could use reservoir sampling (Vitter, 1985) by keeping n vertices and m edges with the minimum hash values in the reservoir to constructed the subgraph. However, additional edges need also to be added to reservoir to count the triangles for T^s_w, $T^s_{(u,v)}$. This is more involved and cumbersome to implement, and for simplicity, is not considered here.

5.2 Case of multiple samples

The algorithms described above can be adapted easily to obtain multiple samples of vertices or edges in parallel, by using independent random walks or multi-hashing functions for the estimation strategy developed in Section 4.

6. Data study

In this section, we assess the performance of proposed sampling and estimation methods for the triangle count distributions on synthetic and real networks. These are summarized in Table 2.

6.1 A single sample

We first consider the Chung-Lu model (e.g. Newman, 2018, Chapter 10), which has the power-law degree distribution $\mathbb{P}(D_{w^*} = k) \sim ck^{-2.5}$, with $N = 20,000$ vertices and $M = 350,000$ edges. In Figure 3(a)–(b), it is assumed that the network has *restricted access*. For Figure 3(a), vertices are sampled with a random walk using the MH version and edges through a standard random walk

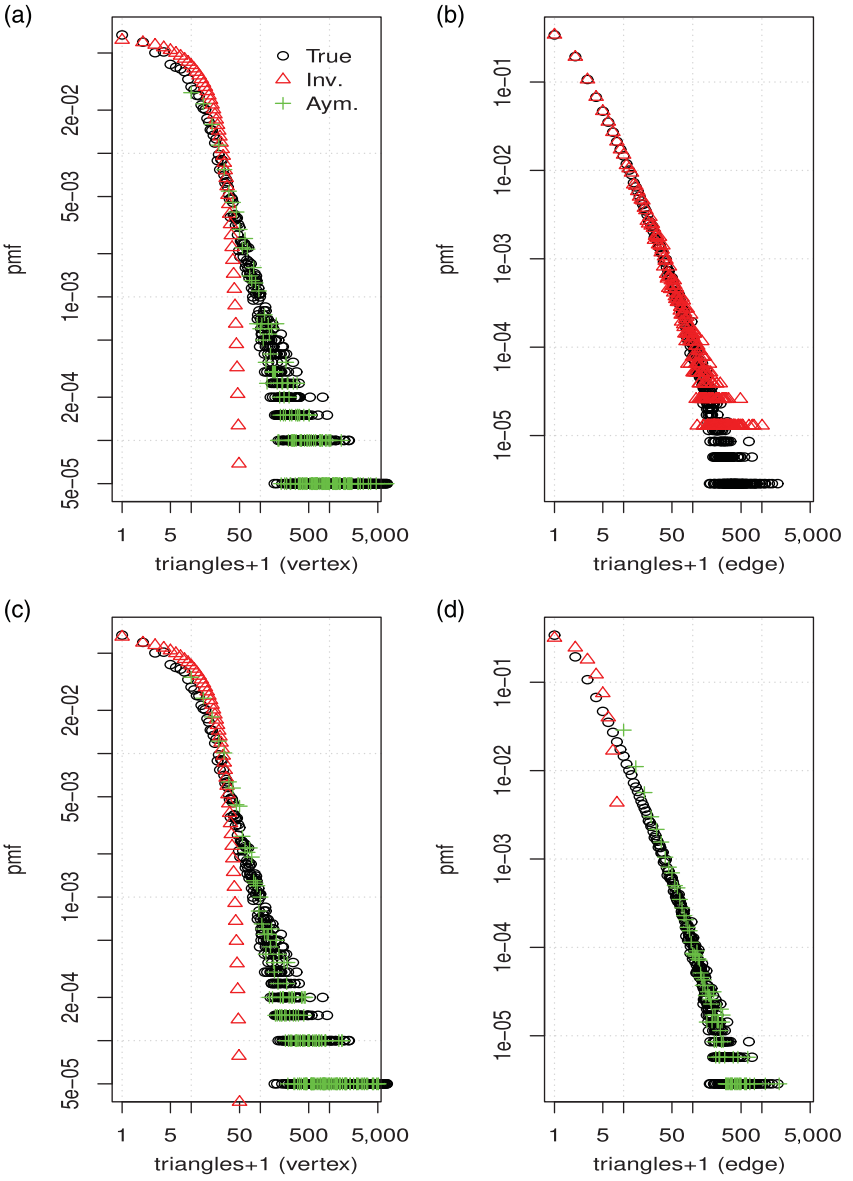

Figure 3. Power-law network: (a) Metropolis–Hastings and standard random walks, (b) Algorithm 1, and (c)–(d) Algorithm 2. The total variation distances are: (a) 0.21, (b) 0.01, (c) 0.19, and (d) 0.23.

(as discussed in Section 5). The sample sizes n and m are equal to 20% of the total numbers of vertices and edges, respectively. We note that typical sampling rates to estimate network distributions, e.g. the degree distribution, are in the range of 10%–30% (Zhang et al., 2015). Figure 3(a) shows the true distribution of triangle counts per vertex and its estimate based on the inversion and asymptotic estimation methods developed in Section 3. With the inversion, the penalized estimator (27) allows to recover well only the bulk of the distribution. The penalization has the effect of shrinking the estimates leading to the reduction of the variance and forces some of the estimates in the tail to be zero. However, the estimation in the tail can be recovered through the asymptotic estimation that scales the sample distribution $\mathbf{t}_v^s$ as in Equation (34).

To compare the estimate with the true triangle count distribution, we use the total variation (distance), which has been previously used in graph sampling (e.g. Mohaisen et al., 2012). It is defined as half of the sum of the absolute differences between two probability mass functions, that is, $d_{TV}(\hat{\mathbf{t}}_v, \mathbf{t}_v) = \frac{1}{2}\sum_i |\hat{t}_v(i) - t_v(i)|$ in our case, and ranges from zero to one. For values of $i \leq k$, the bulk of the distribution $\hat{t}(i)$ is given by the inversion estimation and for $i > k$ by the asymptotic estimation (say, $k = 40$ in Figure 3(a), where the inversion estimation is worse than the asymptotic one). The values of the total variation are given in the captions of the figures.

We have evaluated the impact on the results when the total numbers of vertices and edges are estimated. Assuming the network and access scenario above, where vertices are sampled at random with replacement (in stationary regime), estimators for N and M are given in Katzir et al. (2011) (Section 4) and Kolaczyk (2009) (Section 5.4.2, Equation (5.27)), respectively. We have run the MH algorithm 10,000 times. The average values of the estimates for the numbers of vertices and edges were 20,035.24 and 350,700.60, while the standard errors were 966.50 and 27,128.67, respectively. The averages of the relative errors were 4% (vertices) and 6% (edges). This shows that the estimates are in fact reasonably close to the true underlying values. We checked the estimation of the distribution of triangle count per vertex assuming an error of 10% in the estimates of N and M, such that $\hat{N} = 1.1N$ and $\hat{M} = 1.1M$. The comparison with Figure 3(a) showed still an accurate estimation of the bulk and tail of the distribution. The total variation between the estimate and true distribution was 0.24.

For Figure 3(b), the network is sampled using Algorithm 1 (mixed sampling ($p_v = 1$, m)) where m is equal to 20% of the total number of edges. Since the algorithm emulates $p_v = 1$, the matrix $\mathbf{P}_e$ is not ill-conditioned with the inverse given by Equation (26). In this case, the estimator (23) can be used. Figure 3(b) depicts the estimation of the triangle count distribution per edge when using this inversion. We omit the asymptotic estimation (40) in the plot since it coincides with the inversion estimation from the discussion below Equation (26).

For Figure 3(c)–(d), it assumed that the network is a *streaming graph*. The synthetic power-law network defined above is converted into a stream of edges taken in random order. The network is then sampled through Algorithm 2 with $p_v = p_e = 0.2$. The subgraph G^s selected by the algorithm has 48% of edges and 98% of vertices from the original network. If a two pass algorithm is possible to avoid adding unnecessary edges to obtain the VIE sampling quantities, the sampled graph has 29% and 96% of edges and vertices, respectively. There is a cost in this case of storing more 19% of edges with only one pass of the stream of edges which can not be avoided. The higher number of vertices in G^s is mainly due to the network property $M \gg N$ and $p_e = 0.2$.

The estimation of the distribution per vertex (plot (c)) using the penalized estimator (27) is slightly more accurate when compared to that in plot (a). The asymptotic estimation (34) performs similar in the tail in plot (c). This is due to the fact that with Algorithm 2, edges and vertices are effectively sampled at random (without replacement), while in Figure 3(a) the random walks used to sample vertices and edges at random (with replacement) do so only in the stationary regime. The estimation of the distribution per edge is given in Figure 3(d) using the penalized estimator (27), since now the matrix $\mathbf{P}_e$ is ill-conditioned for small values of p_v and p_e.

We also consider several real-world networks from SNAP database:[1] a collaboration network from the e-print high-energy physics theory (Arxiv HEP-TH) with $N = 8,638$ and $M = 24,806$; a Facebook social network (gemsec-Facebook) with $N = 50,515$ and $M = 819,090$; and an Amazon product co-purchasing network (com-Amazon) with $N = 334,863$ and $M = 925,872$. Figure 4 shows the estimation of the triangle count distributions per vertex and edge for several sampling algorithms dependent on the network access scenario considered, with sampling rates of 20%. The two estimation methods show that the true triangle count distributions can be recovered quite accurately which agrees with the results for synthetic networks.

Finally, we comment on the relations between power-law exponents (Section 3.2.3). For the synthetic network, the estimated exponent of the triangle count distribution per vertex is $\beta = 0.73$

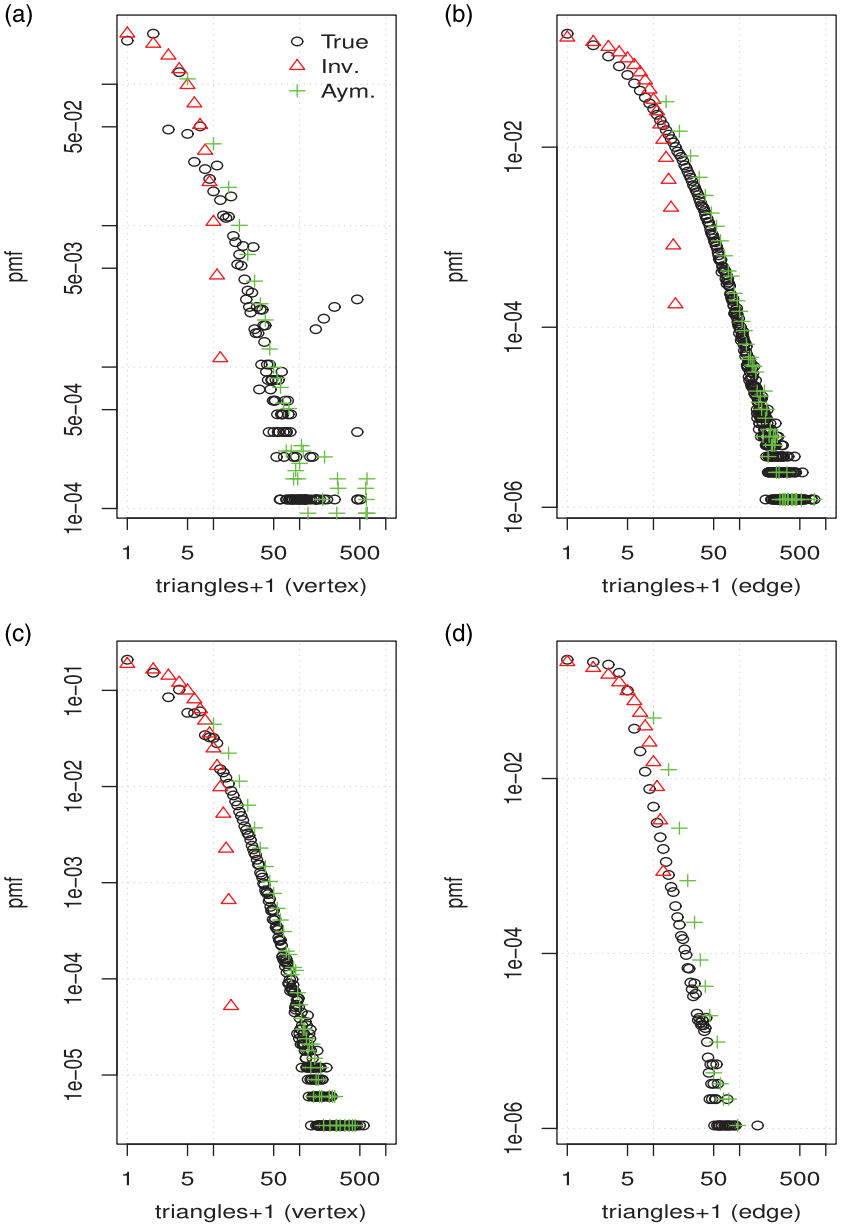

Figure 4. (a) Arxiv HEP-TH (Metropolis–Hastings and standard random walks), (b) gemsec-Facebook (Algorithm 2), (c), and (d) com-Amazon (Algorithms 1 and 2, respectively). The total variation distances are: (a) 0.26, (b) 0.24, (c) 0.18, and (d) 0.29.

(using the maximum likelihood estimation in the formula (10.9) of Newman, 2018) while for the degree distribution, the exponent is $\alpha = 1.5$ which agrees with Equation (45). For the com-Amazon network, we found $\beta = 1.24$ and $\alpha = 2.28$.

6.2 Multiple samples

For the com-Amazon network in a restricted access scenario, we evaluate the estimation method of Section 4. Vertices are sampled using the MH random walk and edges with r independent

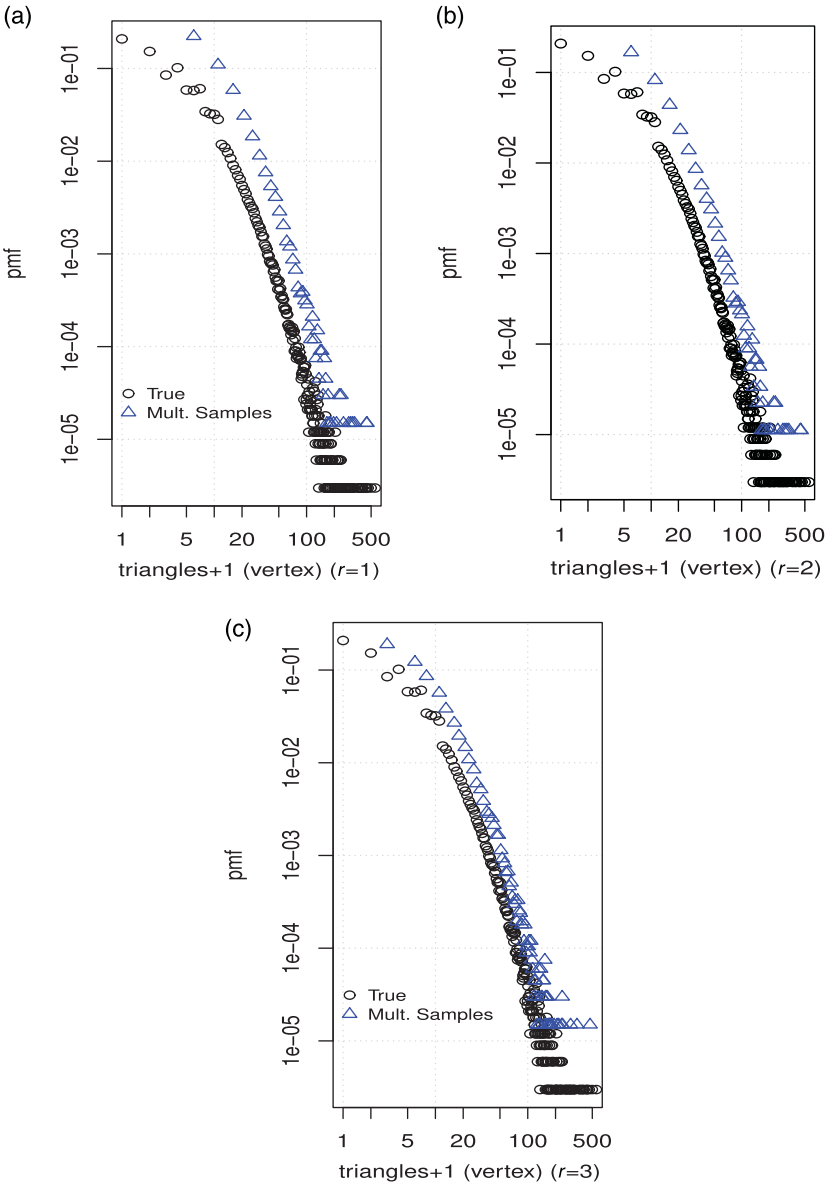

Figure 5. Com-Amazon: Metropolis–Hastings and standard random walks. The total variation distances are: ($r = 1$) 0.68, ($r = 2$) 0.53, and ($r = 3$) 0.32.

standard random walks in parallel. Figure 5 shows the estimation of the distribution of triangle counts per vertex for $r = 1, 2, 3$. The size of each sample is 20% of the total number of the units. With only one sample of edges ($r = 1$), the distribution is overestimated when using the estimator (56). Increasing the number of samples of edges obtained in parallel, the bias of the estimation decreases, however, at higher sampling cost (with $r = 3$ effectively corresponding to sampling 60% of the edges). For $r = 3$, the performance is still worse than when using the inversion and asymptotic estimation with just one sample (Figure 4(c)). Similar findings hold for the estimation of the distribution of triangle counts per edges (Section 4.2).

7. Conclusions

In this paper, we focused on the triangle count distributions of vertices and edges in networks, and their estimation through a newly introduced sampling method (VIE sampling). Three sampling designs of VIE scheme were proposed motivated by common network access scenarios. We developed an estimation method based on inversion for the bulk of the distribution and an asymptotic method for the tail. We also proposed a single approach to estimate the entire distribution using multiple VIE samples collected in parallel. For the several network access scenarios, algorithms were presented to sample the network using random walks in the restricted access network scenario and using hashing in the setting of streaming edges. The proposed estimation methods were evaluated on several synthetic and real-world networks, showing a satisfactory performance for the inversion and asymptotic approach with a single sample, with a higher cost for the multiple samples approach.

Several open questions were already raised for future work. For example, one open problem concerns the relation between the power-law exponents of the degree distribution and the triangle count distribution per edge (see Section 3.2.3). In other directions, one could possibly consider graphs with repeated edges (multigraphs) or directed graphs, and count distributions per vertex and edge for higher order graphical structures other than triangles such as k-cliques.

Acknowledgments. The authors would like to thank the editor and three anonymous reviewers for their many useful comments that led to a significant improvement of the paper.

Conflict of interest. None.

Note

1 https://snap.stanford.edu/data/index.html.

References

Al Hasan, M., & Dave, V. S. (2018). Triangle counting in large networks: A review. *WIREs Data Mining Knowledge Discovery*, *8*(2), e1226.

Antunes, N., Guo, T., & Pipiras, V. (2020). Induced edge samplings and triangle count distributions in large networks. In H. Cherifi, S. Gaito, J. F. Mendes, E. Moro, & L. M. Rocha (Eds.), *Complex networks and their applications VIII* (pp. 203–215). Springer International Publishing.

Antunes, N., & Pipiras, V. (2016). Estimation of flow distributions from sampled traffic. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, *1*(3), 11:1–11:28.

Bar-Yossef, Z., Kumar, R., & Sivakumar, D. (2002). Reductions in streaming algorithms, with an application to counting triangles in graphs. In *Proceedings of the 13th Annual ACM-SIAM SODA* (pp. 623–632).

Becchetti, L., Castillo, C., Donato, D., Baeza-Yates, R., & Leonardi, S. (2008). Link analysis for web spam detection. *ACM Transactions on the Web*, *2*(1), 2:1–2:42.

Buriol, L. S., Frahling, G., Leonardi, S., Marchetti-Spaccamela, A., & Sohler, C. (2006). Counting triangles in data streams. In *Proceedings of the 25th ACM SIGMOD-SIGACT-SIGART PODS* (pp. 253–262).

Eckmann, J., & Moses, E. (2002). Curvature of co-links uncovers hidden thematic layers in the world wide web. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(9), 5825–5829.

Eldar, Y. C. (2009). Generalized SURE for exponential families: Applications to regularization. *IEEE Transactions on Signal Processing*, *57*(2), 471–481.

Jha, M., Seshadhri, C., & Pinar, A. (2015). A space-efficient streaming algorithm for estimating transitivity and triangle counts using the birthday paradox. *ACM Transactions on Knowledge Discovery from Data*, *9*(3), 15:1–15:21.

Katzir, L., Liberty, E., & Somekh, O. (2011). Estimating sizes of social networks via biased sampling. In *WWW'11*. ACM.

Kolaczyk, E. D. (2009). *Statistical analysis of network data*. New York: Springer-Verlag.

Leskovec, J., & Faloutsos, C. (2006). Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD'06 (pp. 631–636).

Lim, Y., Jung, M., & Kang, U. (2018). Memory-efficient and accurate sampling for counting local triangles in graph streams: From simple to multigraphs. *ACM Transactions on Knowledge Discovery from Data*, *12*(1), 4:1–4:28.

Mohaisen, A., Luo, P., Li, Y., Kim, Y., & Zhang, Z. (2012). Measuring bias in the mixing time of social graphs due to graph sampling. In *IEEE Military Communications Conference, MILCOM 2012* (pp. 1–6).
Newman, M. (2018). *Networks: An introduction* (2nd ed.). New York: Oxford University Press.
Palla, G., Derényi, I., Farkas, I., & Vicsek, T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, *435*(7043), 814–818.
Stefani, L. D., Epasto, A., Riondato, M., & Upfal, E. (2017). TRIÈST: Counting local and global triangles in fully dynamic streams with fixed memory size. *ACM Transactions on Knowledge Discovery from Data*, *11*(4), 43:1–43:50.
Thompson, S. K. (2012). *Sampling* (3rd ed.). Wiley Series in Probability and Statistics. Hoboken, NJ: John Wiley & Sons, Inc.
Tillé, Y. (2006). *Sampling algorithms*. Springer Series in Statistics. New York: Springer.
Tune, P., & Veitch, D. (2011). Fisher information in flow size distribution estimation. *IEEE Transactions on Information Theory*, *57*(10), 7011–7035.
Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, *11*(1), 37–57.
Zhang, Y., Kolaczyk, E. D., & Spencer, B. D. (2015). Estimating network degree distributions under sampling: an inverse problem, with applications to monitoring social media networks. *The Annals of Applied Statistics*, *9*(1), 166–199.

Cite this article: Antunes N., Guo T. and Pipiras V. (2021). Sampling methods and estimation of triangle count distributions in large networks. *Network Science* **9**, S134–S156. https://doi.org/10.1017/nws.2021.2