Review Response: “Adaptive Shrinkage Estimation for Streaming Graphs"

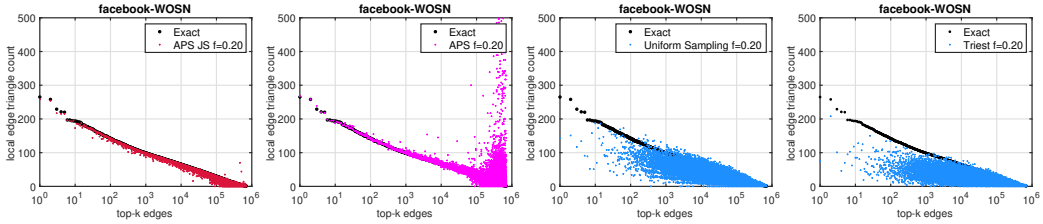
We thank all the reviewers for their time and feedback. The main reservations of the reviewers are addressed below, and all requested clarifications will be added to the final version.

Citations & references (Reviewer #2)

Thank you for pointing us to this work, we completely agree, and have now included references to the works by Ledoit & Wolf, 2003; Schäfer & Strimmer, 2005; Chen et al., 2010; Xu et al., 2014.

Demonstration on a timestamped graph (Facbeook WOSN 2009) (Reviewer #2)

Thanks for pointing us to this. As suggested, we provide a demonstration on a timestamped graph.



Spectral error: APS JS (with Shrinkage): 0.04, APS: 0.31, Uniform Sampling: 0.40, Triest: 0.58.

How is ϕ (line 4 of algo 1) chosen, and what is the impact of different choices (Reviewer #4)

We use ϕ in Alg 1 to denote the initial weight of an edge, such that $\phi > 0$. This initial weight is required to guarantee that each edge is assigned a non-zero probability. We chose $\phi = 1$ to be comparable with the edge weight increments due to subgraphs incident to each edge. This procedure allows edges to have a chance to be included in the sample with a non-zero probability, regardless of the number of subgraphs incident to them, but not so large as to damp out their topological weight.

Comparison with Triest is nice, but it is unclear which version of Triest used. (Reviewer #4)

We compare with Triest-IMPR [40], the improved Triest algorithm with higher estimation quality.

Highlighting Algorithm & Complexity Analysis (Reviewer #3)

- Reviewer #3: “The running time depends on the number of triangles in the graph clearly, but each triangle might be computed multiple times."

There may be some confusion here. The algorithm uses a *small fixed-size memory* to store the sample ($m = |\widehat{K}|$). The first m edges are added to the sample. Then, each subsequent edge is *provisionally* included in the current sample from which an edge is discarded along with its associated variables (i.e., to maintain the sample size m). When a new edge arrives in the stream, it is processed *only once*, and each motif completed by the new edge whose other edges are in the current sample is also processed only once. See Sec 2.3 (line 202) for the cost analysis of all sample updates (insert/delete/update).

Emphasizing Algorithm Guarantees (Reviewer #3)

The algorithm is guaranteed to provide the unbiasedness property for the reasons stated below.

- Reviewer #3: “The estimator is unbiased only for edges in the sample."

There is some confusion. Edges are included in the sample based on their assigned probabilities. Thus, edges not retained in the sample indeed have zero estimator. Edges that are included in the sample have an estimate which is reciprocal of their probability to be retained in the sample. Unbiasedness is a general property of inverse probability estimators that are used in numerous areas of computer science & statistics, including graph sampling; e.g. reference [1] in the paper. This procedure guarantees unbiased estimators for the entire graph (i.e., $\mathbb{E}[\widehat{\mathrm{A}}] = \mathrm{A}$, A denotes adjacency matrix).

- Reviewer #3: “You need some sort of analysis about the probability an edge stays in the sample."

Yes, this is precisely what we do (see Sec 2.2): for each edge in the sample, we compute the probability that it has been retained so far (Alg 1, line 11); the inverse probability estimate for that edge is the reciprocal of that probability. Theorem 1 says that the probability of a subgraph is the product of the probabilities of its edges (computed in Alg 1, line 13), and finally, the corresponding unbiased estimator for that subgraph is the reciprocal of that product, which gets added into the estimated count of subgraphs incident to edges (Alg 1, line 14). Theorem 1 states the inverse probability estimator for subgraph counts and proves it is unbiased (see lines 165-167 for details). It shows how the estimators can be computed in practice from random/state variables during stream processing.

- Reviewer #3: “Aren’t the weights always non-decreasing?"

Theorem 1 applies for any weights, non-decreasing or not. In our case, the weights are non-decreasing, and Theorem 2 shows the computational benefit results from this property, which we exploit in Alg 1.

Typos/Presentation (Reviewer #2 & Reviewer #4)

Last, we thank all reviewers for pointing out a few typos and have now fixed them. Reviewer #2 & #4 suggested to move a few parts to supplementary materials. As suggested, we will move the derivation of the optimal shrinkage coefficient λ in section 3.1, and Section 2.3 to the supplementary materials.