Modified k-core measures for increased granularity

Yixuan (Lisa) Shen[a,1,2], **Daniel Luo**[a,1], **Pratyusha Majumder**[a,1], **and Dhruv Chakraborty**[a,1]

[a]University of California, Los Angeles, CA 90095

This manuscript was compiled on January 9, 2021

K-shell decompositions offer an extremely fast and interpretable way to analyze the structure of a network. The resulting core-periphery structure is increasingly studied given its ability to identify influential nodes in a wide variety of applications. We improve the somewhat limited scope of the algorithm by adding random variations that result in more granular results. In particular, our modifications strictly increase the number of resulting shells and spread the nodes in a more even distribution. We analyze the differences between the core decomposition algorithm and our modified versions on several types of theoretical and empirical networks, finding that our version best applies to those with power-law degree distributions. Moreover, we observe statistically significant increases in key metrics such as degree, betweenness and eigenvector centrality for nodes placed in the highest k-shell when comparing the modified versions with the regular algorithm - suggesting that a few nodes were correctly pushed to higher shell status by our modified versions. Although we found our algorithms had fairly low variance in the core they assigned each node too, we think they could be furthered to achieve better results if more deterministic processes were to underlie the node selection process.

k-cores | centrality | networks

A k-core is a connected set of nodes where each node is adjacent to at least k others in the set, and a k-shell is defined as the set of all nodes belonging to the k^{th}-core but not to the $(k+1)^{th}$-core (1). We can find these cores quickly even in large networks by simply iteratively removing nodes that have degree less than k until every node has degree at least k. This process leads to the k-shell decomposition (also called k-core decomposition) of the network, analyzing which is useful to identify the core and periphery of the network (2).

K-shell decompositions are useful since they not only provide a fast algorithm to study a network's structure, but also guide us to nodes that are, in some sense, central to the network. For example, in "Identification of Influential Spreaders in Complex Networks," Kitsak et al. (3) argue that k-shell decomposition identifies key spreaders in networks more accurately than looking at the most connected individuals, with a particular focus on networks studying the propagation of infectious disease or information. In particular, if carriers of infectious disease are present in higher cores, they are more likely to be an influential spreader in the network. One of the key arguments made is that when considering multiple spreaders, the distance between them is a crucial factor for studying the extent of propagation. This distance is decreased the higher the k-shell spreaders belong to, since each node then has degree at least k in the network.

Furthermore, in "The Critical Periphery in Growth of Social Protests," Barberá et al. (4) show that the periphery of a network is critical to increasing the number of people exposed to the protest information, primarily through the use of various Twitter retweet networks. In doing so, they also find that the so-called core of the network is responsible for producing and starting the spread of novel information itself. Essentially, while the periphery is crucial to improving the reach of a network, the core is in charge of much of the activity. The authors look at several protest networks and even a network of discussions about the Oscars, and we present their results for the 2013 Taksim Gezi Park protests in Turkey in figure 1 below. A key observation is that while lower k-shells (periphery nodes) have a larger number of people in this network, graph B show us that in both cases far more tweets are produced by higher k-shells. Therefore, if we are concerned with nodes that are responsible for the start of the spread of information itself, higher k-shells are a good identify them.

While we find it important to consider the k-core structure of a network due to the above-mentioned reasons, we find k-shell decompositions have some limitations. One primary drawback with the method is that nodes are simply not separated enough. There are not enough shells, and even in cases when this is not the problem, nodes are not well-distributed across these shells. Figure 2 below shows us these distributions of nodes into k-shells for an Erdos-Renyi graph sampled from $G(n = 1000, p = 0.25)$ and the Caltech Facebook friend network $(N = 769, E = 16,656)$ (5). We can see clearly that most of the nodes are placed in the highest k-core, which exacerbates the issue since we are mostly concerned with the differences in these "more central" nodes. Moreover, when sampling from Barabasi-Albert random graphs (6) that obey the power-law degree distribution structure seen in many real world networks, we found that often all nodes were dumped in a single core. Thus, one of the primary modifications we make

Significance Statement

Although important nodes in networks are typically found through centrality measures, k-shell decompositions offer a fast algorithm for splitting nodes into interpretable cores that can also be used to study the structure of a network more broadly. Such decompositions have been of interest more recently due to their applications in identifying crucial nodes in networks modelling information and disease spread. We modify the k-core algorithm with random variations in node selection processes such that nodes are spread more evenly over a greater number of shells, offering more granularity. In addition, we find that although the processes underlying our algorithm are random, only specific nodes are offered higher k-shell status through our algorithms, particularly those with higher degree and betweenness centrality.

L.S. worked on code and analyzing data/results for edge-adding method; D.C. worked on initial research, code for saving nodes and code for analyzing data; D.L. worked on code for saving edges and analyzing results for saving nodes; and P.M. worked on analyzing data. L.S. , D.C., D.L., and P.M. designed research and wrote the paper.

The authors declare no conflict of interest.

[1]Authors contributed equally to this work.

[2]To whom correspondence should be addressed. E-mail: yshen17@g.ucla.edu

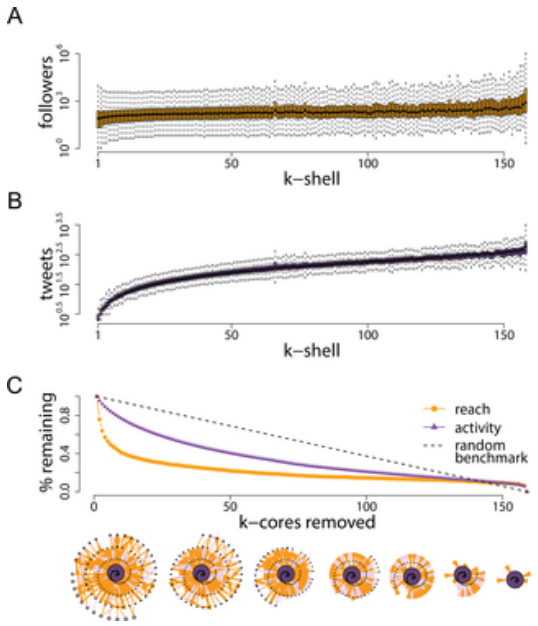

Fig. 1. Audience size and activity levels across k-cores at 2013 Taskim Gezi Park protests in Turkey. https://doi.org/10.1371/journal.pone.0143611.g004

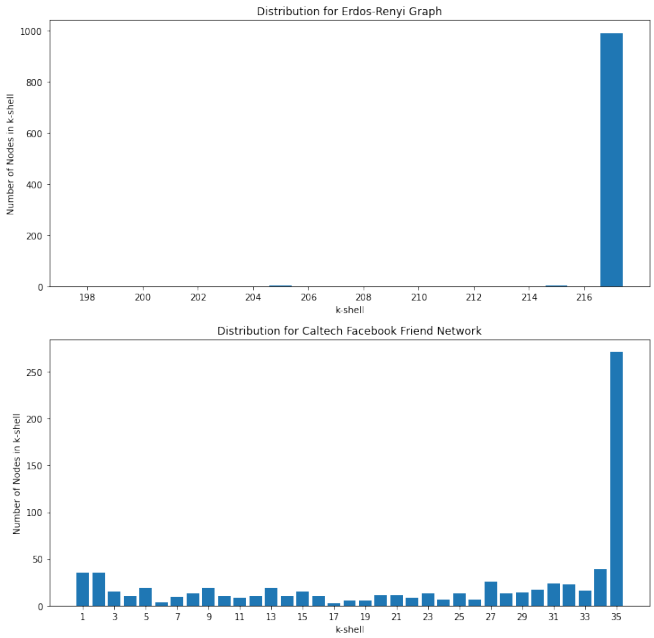

Fig. 2. Distribution of nodes into respective k-shells for Erdos-Renyi Graph and Caltech Facebook friend network

to the k-core algorithm is to separate the nodes into more shells and even the distribution of the nodes in a meaningful manner, such that we offer more granularity in differentiating the nodes.

Perhaps more interestingly, we are also concerned that the k-core algorithm is disadvantageous to nodes that belong to what we think of as the "outer core" of the network. These are unduly punished by the removal of the periphery, as these connections are not considered as the algorithm iteratively reaches the k-core level when these nodes are removed. In particular, we believe that differences among core nodes are largely ignored since such a large number of nodes are placed into the highest k-shell. By our intuition, the so-called outer core should actually have an even higher k-shell number, since it is not only highly connected to the internal core of the network, but also connects this so-called internal core to the periphery. In particular, upon removal of the periphery, these nodes' degrees are iteratively reduced while the nodes belonging to the internal core are not disadvantaged in this manner. In this sense, we would expect an increase in the betweenness centrality in the highest k-shell our modified algorithm identifies in the network.

We thus improve the k-core measure by spreading out the obtained shells further, more evenly assigning nodes to cores, and capturing relevant differences using our modified measures for more connected nodes, particularly for those in the highest shell. We broadly modified the k-core algorithm in two ways: saving nodes at each iteration or adding edges at each iteration. For saving nodes, at each iteration that removes nodes to find the k^{th}-core, we randomly save nodes being removed with some probability p_s. We also tried an approach to "save an edge" via sampling over the nodes' degrees and saving a single edge with the node instead of the node generally as above, but we found this did not work and gave us largely same results as the k-core algorithm. For adding edges, we add some portion p_a of edges not present in the sub-network considered at each particular iteration, in hopes to make up for edges lost earlier in the process.

Results

Saving Nodes Method. We can first start to discuss the results from the node saving k-core method. We tested it on several different types of graphs to see whether the results from these methods are either irrespective of the type of graph that the method is implemented on or if the method has a bias towards certain types of graphs.

Thus, it was tested on a Watts-Strogatz random graph to see how it performs on a graph with high local clustering but no power-law degree distribution (7). It was then tested on a Holme-Kim random graph, which follows the same process as Barabasi-Albert graphs where a new node A attaches itself to a node B in the existing network with preferential attachment for one edge, then A attaches itself to one of B's neighbors with probability p_{HK} for the remaining m-1 edges (8). This creates a graph with clustering and a power-law degree distribution which does not sort all nodes into single shell like Barabasi-Albert. Lastly, the node saving method was tested on the Caltech Facebook friendship network (5). This allowed us to see how our method performs on empirical networks.

The Watts-Strogatz network we used was constructed with 1000 nodes, each node connected to 250 of its neighbors and the probability of rewiring is 0.25. This given network is dense with high clustering. When applying our node saving modified k-core method to this Watts-Strogatz network, we found that it was unable to create much spread and granularity in the higher level shells. When $p_s = 0.1$ and 0.5, which represents the probability that a node at each step of the k-core method gets saved, essentially all the nodes get sorted into a single shell with the result being pretty much identical to the normal k-core algorithm, only that the largest shells were shifted right a few levels (See Figure S2). We can however set the p_s to 0.9

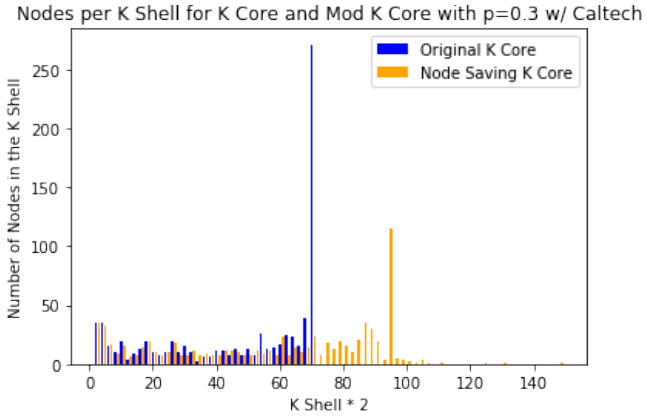

Fig. 3. Node-saving method: Increase in spread in higher level shells with modified method for Caltech network *[Please see supplementary information for additional figures]*

and introduce a huge amount of randomization which helps us reach our desired results of spread in the upper level shells, but this would cause other undesired effects that will be discussed later in the discussion of choosing an optimal parameter p_s. In terms of correlation with centrality measures, we did find that there is a slight increase in the centrality measures (degree, eigenvector, betweenness) for Watts-Strogatz graphs in the highest shells with the node saving k-core method compared to the original k-core method (See Figure S5).

Next, we applied the same technique to a Holme-Kim random graph with 1000 nodes where each additional node is connected to 30 existing nodes with preferential attachment, and the probability of creating a triad is 0.5. We can see that even when p_s, the probability of saving a node, is quite low set at 0.1, there is a strong spread in the distribution of nodes into upper-level shells, which fulfills our goal of creating a higher granularity importance measure with k-cores (See Figure S3).

We also observed an increase in centrality measures in the highest level shells resulting from the node saving k-core when compared to the original k-core (See Figure S6). We confirmed this statistically specifically in the last core by running our algorithm 100 times to get a sample of the degree, eigenvector and betweenness centrality for the nodes in the last core. We compared this to the centrality measures with the normal k-core algorithm and saw a statistically significant increase (alpha $<$ 0.05) in the centrality using our modified k-core algorithm.

Finally, we can examine the results from applying our node saving method to the Caltech network. We saw that there was a slight increase in the spread in the distribution of nodes into higher level shells at p=0.1. It reduced the number of nodes in the last shell by around a quarter and placed those nodes into the highest shells. Even greater spread was achievable by raising p_s to 0.3 (see Figure 3). At this point there is a strong level of spread in the upper cores which yields greater granularity to understand which nodes are more important in the network (See Figure S4).

Additionally, we saw that the higher level cores with the node saving method on average yielded nodes with greater degree, eigenvector and betweenness centrality than the original k-core. This further confirms that these nodes are likely to be influential nodes in the network (See Figure S7).

Adding Edges Method. Similar to our node-saving method, we first tested this method on three types of well-known generative networks: an Erdos-Renyi random graph, Watts-Strogatz small-world network, and a Barabasi-Albert network(9)(7)(6). Then, based on the intuitions we get from these three graphs, we tested the method on the Caltech Facebook friendship network to ensure its validity on a real-life social network(5).

By adding 1% of edges in the remaining network (p_a = 0.01) at each step of k-core decomposition, the modified k-core measures achieve the desired high levels of granularity for the Erdos-Renyi random graph (1000 nodes and an edge exists with probability = 0.25) and the Watts-Strogatz small-world network (1000 nodes, mean degree = 250, rewiring probability = 0.5). In comparison with the original k-core measure, there are fewer numbers of nodes in the highest shell, and the mean centrality measures of nodes in the top five k-shells–including degree, betweenness, closeness, eigenvector, PageRank centrality–are also statistically significantly higher than the corresponding top five k-shells respectively at 5% significance level ($p < 0.001$). We yield similar results with varying parameters of Erdos-Renyi and Watts-Strogatz models of similar size. The results from applying the edge-adding methods on these two models are promising (see Fig. S9). In alignment with our goal, the modified k-core method sorts nodes into more levels of k-shells in a meaningful manner.

One observation from the tests on Erdos-Renyi graphs and Watts-Strogatz graphs is that both our modified k-core measure and the original k-core measure yield similar mean degrees of nodes in the highest shell, which are slightly higher than the expected mean degrees of nodes of the graphs based on their parameters. Although our modified k-core measure performs slightly better with a statistically significant increase in mean degrees with reduced variance and other centrality measures in the highest shell, the difference is rather small. We propose that the network structures of these two graphs that generate a large number of nodes with similar degrees may account for the reason behind this observation. Thus, we further conduct analysis on the Barabasi-Albert network, which generates a power-law degree distribution that more closely resembles empirical networks.

Implementing the edge-adding method with 1% addition of possible edges to a Barabasi-Albert network (i.e. 1000 nodes, 51 nodes present in the beginning with each new node connecting to 50 other nodes in the network) at each step of k-core decomposition, we have again observed an increase in the numbers of k-shells with our modified k-core measure (see Table S1). For Barabasi-Albert models, the original k-core measure gives very few shells, and most of the time results in only one 50-core with all the nodes. The edge-adding k-core measure extracts around a third of the size of the 50-core from the original method at the highest k-shell. The nodes in the highest k-shell have significantly higher mean degrees and other mean centrality measures in comparison with the original 50-core. The result demonstrates that our modified k-core measure meaningfully sorts the nodes into more levels of k-shells in comparison with the original k-core method.

With the success of our exploration of the edge-adding method in Barabasi-Albert generative models, we apply our modified k-core algorithm with 1% addition of edges to the Caltech Facebook friendship network (5). The results align well with the results we get from the generative models. There is an

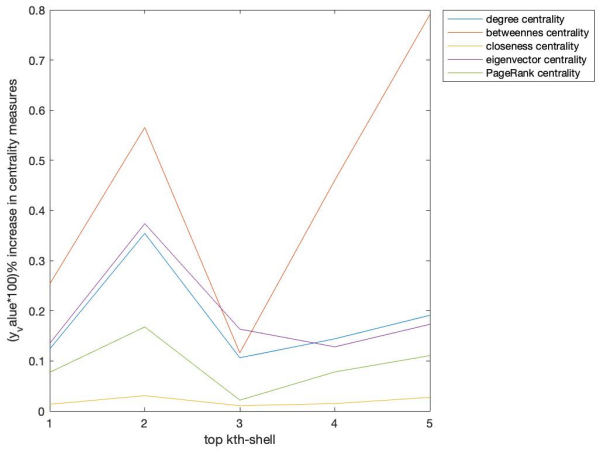

Fig. 4. Edge-adding method: percentage increase in centrality measures in top five k-shells *[Please see supplementary information for additional figures]*

increase from 35 to 42 k-shells with around 30% reduction in numbers of nodes at the highest levels of k-shell (see Fig. S10). We conducted a t-test on the top five levels of k-shells to assess the differences in centrality measures and our modified k-core measure yielded a statistically significant increase across all centrality measures ($p < 0.001$) over 100 runs at 5% significance level. Figure 3 illustrates the percent increase in centrality measures across top five shells using our edge-adding k-core method in one run (see Table S2 for details).

As our modified measure entails randomly adding edges in the network, we want to ensure that our method is consistent with regards to sorting nodes into k-shells. Specifically, the method is valid if it consistently sorts a specific node in the network into a specific k-shell. Therefore, we have tested the consistency by running our method 100 times on the Caltech Facebook network and analyzed the variance of k-shells a node is placed into for each of 769 nodes across 100 runs. With adding 1% of edges, the mean standard deviation is 0.304 and the highest standard deviation observed for only a small number of nodes is less than 1.41 (refer to Fig. S13). The analysis of variance illustrates that our algorithm is consistent in sorting nodes into k-shells, proving that the granularity and meaningfulness achieved are not random.

Discussion

Optimizing p_s and p_a. From the aforementioned results from applying the node saving k-core method to various types of graphs, it became apparent that p_s plays a central role in getting the desired results of increased spread in the distribution of nodes into higher level shells. We also observed that setting a high p_s will create a huge amount of spread in higher level shells but at the cost of increasing the variance of which shell a node gets sorted into.

This second type of increased variance in terms of which shell a node gets sorted into is ultimately undesired because when it is too high, there is instability with this k-core method. It would be undesirable to have different runs of the same modified k-core method on the same network yield vastly different results for which nodes should be important and which nodes should be unimportant. As p_s approaches 1, the algorithm picks off nodes at random from the outside which makes the results fairly meaningless (See Figure S12).

Thus, there is a trade-off between the desired spread in higher level shells and the undesired inconsistency of results. Because of this, it becomes necessary to choose an optimal p_s that will yield the best results. One possible approach would be to quantify the spread in the highest shells, potentially through a standard deviation metric. Then create a metric to measure the variance in which shell a node gets sorted into. The goal is to choose a p_s to maximize the first quantity and minimize the second quantity. Another way to consider this is to maximize the ratio between the first and second quantity.

In addition, choosing a p_s that satisfies our goal of increased spread in higher level k-shells typically also satisfies our goal for increased centrality in the higher level k-shells. Thus from our findings we do not need to be concerned for choosing a p_s that specifically maintains an increased centrality.

Similar to the node-saving method, our edge-adding method also includes a parameter p_a, which is the percentage that determines the number of edges added into the remaining network at each step of k-shell decomposition. Specifically, after each round of removing nodes, we add in p_a percent of the total number of possible, non-existent edges (total number of edges that could exist minus already-existing edges) between node pairs that are previously not connected. The feasible value of p_a ranges from zero to one. However, we are wary of the fact that this modification of k-core measures by adding edges that do not exist in the original network is equivalent to creating nonexistent connections between entities. Therefore, the value of pa is kept under 2% for a network consisting of around 1000 nodes to avoid a significant modification of the original network that may make the results, although more significant, un-interpretable.

To see how varying parameter p_a may influence the results of edge-adding method, we have tested $p_a = 0.005$, 0.01, 0.02 on the Caltech Facebook network (refer to Fig. S14). With increasing p_a values, we can observe that the modified k-core measure achieves a higher granularity with more increase in centrality measures in the highest shell. Nonetheless, the assessment of the variance of which shell each node is sorted into over 100 runs shows that a larger p_a value, although does not generate a higher mean variance, results in a higher maximum of variance in a small number of nodes. This is undesirable because a higher variance indicates that the assortment of nodes is not consistent, contradicting with our purpose of creating a method that sorts nodes into different levels of shells meaningfully based on its importance as indicated by centrality measures.

As the algorithm makes p_a sensitive to both the size and edge density of the network, future analysis should be conducted to optimize value of p_a based on size and sparsity of the network to achieve the best results for k-shell decomposition with minimal adjustment to the original network.

Effectiveness on Particular Networks. Regarding the performance of our modified k-core method on different networks, we found that our method was effective with the Holme-Kim random graph as well as the Caltech Facebook network but was relatively ineffective with the Watts-Strogatz random graph. This shows that there are certain properties in networks that favor our methods to reach our goals but we need to conduct further analysis to understand what these exact properties

are.

One thing to notice is that Holme-Kim graphs and the Caltech network both have power-law degree distributions and yield a greater spread in the distribution of nodes into k shells even at relatively low levels of p_s. One reason could be that these types of graphs have a periphery of nodes outside of the central core. As the modified k-core algorithm trims nodes in this periphery, some of them are randomly saved. This process pushes these nodes into a higher shell as well as its neighboring nodes deeper inside the core into a higher shell as well. This creates the spread that we see in the highest level k-shells in the Holme-Kim graph and the Caltech network.

This is in contrast to Watts-Strogatz graphs that are highly clustered and usually don't possess a distinct periphery. This means as we gradually trim away the outer shells of the network while saving a few nodes at random, the final result is not that different from the normal k-core method at low to medium levels of p_s because the chances of saving the same node or a the same node's neighbor in the core is relatively low.

Another way to think about this is we can imagine Holme-Kim and the Caltech networks as being balls with spikes in them whereas the Watts-Strogatz graph is a uniformly round ball. The normal k-core algorithm shaves the outside of each to find each shell while our modified k-core misses a few spots at random. If you apply the modified k-core to the spikey ball, and those saved spots coincide with the spikes, it creates variance at the end of the algorithm since the spikes stay around for longer. Compare this to missing spots on the uniform ball. Those missed spots get trimmed away soon after which leads to the lower variance in the distribution of does into shells at the end of the algorithm.

These results are beneficial to our end goal since we want to apply this algorithm ultimately to social networks that share similar structure with Holme-Kim and the Caltech network. If this intuition is correct, our node saving method would be able to put nodes that connect the core with the periphery into one of the higher level cores which would help us identify these nodes that are potentially important in cases involving social networks. For example, these nodes could be the people on Twitter that are part of the core community that generates information but also spreads the news to other users. This is an improvement on the normal k-core algorithm that does not let us identify these nodes since it grants lower spread and granularity in higher level cores.

The modified k-core measure by adding edges succeeds in increasing granularity of k-shell decomposition in a meaningful way across all types of the graphs tested with a small value of p_a, chosen according to earlier discussion about p_a. Nonetheless, edge-adding method seems to achieve the same goal of meaningful granularity in different ways on networks with power-law degree distributions (Barabasi-Albert network and Caltech Facebook social network) compared to networks consisting large numbers of nodes with similar degrees (Erdos-Renyi graph and Watts-Strogatz graph).

For networks with a power-law degree distribution, there is a significant reduction in the numbers of nodes at the highest level of shell with our edge-adding method, ranging from around 30% for Caltech Facebook network to 60% for Barabasi-Albert graphs. The differences in mean centrality measures in higher-level shells are statistically significant. Therefore, if we consider the core of the network to be the highest shell achieved by the original k-core methods, this suggests that the edge-adding method meaningfully spread nodes in the core from the original k-core method into more levels of k-shells. Thus, for networks with power-law degree distributions, the edge-adding method increases granularity by dividing nodes in the core into more levels based on the nodes' importance, aligning well with our goal to achieve a better understanding of the core with higher granularity and its correlation with higher centrality measures at higher-level shells.

For Erdos-Renyi and Watts-Strogatz networks, as discussed in the aforementioned results section, the mean degree of nodes in the highest shell obtained by the modified k-core method is higher but still similar to the measure achieved by the original k-core method, with both being close to the mean degrees of nodes in the entire network. This is expected as most of the nodes in the network share similar numbers of degrees. In comparison with the performance of the edge-adding method on networks with power-law degree distribution, the reduction in the numbers of nodes and increase in mean centrality measures for nodes in the highest level of shell is statistically significant but not as notable. Therefore, if we again consider the core to include all nodes in the highest shell achieved by the original k-core method and the other nodes as the periphery, the edge-adding method preserves a slightly better core with similar properties and increases granularity by sorting the nodes in the periphery into more levels. This is different from our desired goal, but it suggests that peripheral nodes may also have a stratification of importance in a similar way as what we have proposed for the core nodes.

We propose that the difference in performance of our modified k-core measure by adding edges is a result of the setup of our edge-adding algorithm. Since our algorithm of adding edges is partially in favor of nodes with lower degrees, as the edges are added between a node chosen uniformly at random and another node that does not already have a connection with previously chosen node, it would yield different results for networks with most of the nodes sharing similar degrees (i.e. Erdos-Renyi and Watts-Strogatz network) and networks consisting of a large number of low-degree nodes and a few high-degree nodes (i.e. Barabasi-Albert network and Caltech Facebook network). However, further analysis needs to be conducted in order to fully understand the reasons behind this difference generated and its implications. For instance, we can examine the patterns of which exact nodes are sorted into each level of shells and extend our edge-adding methods to graphs with other patterns of degree distributions.

Further considerations. To further our discussion above, we would like to test our modified k-core methods on other models and empirical data with power-law degree distributions to see if our results are representative of our methods when applied to models of that distribution in general. In terms of generative models with power-law degree distributions, the node-saving method was only tested on the Holme-Kim model with high clustering ($p = 0.5$), so we would like to explore generative and empirical networks with low clustering to see if our conclusions can be extended to these networks. The networks we tested our methods on had 769-1000 nodes so testing them on smaller and also larger networks, and varying parameters used to construct the generative models could give us a better understanding of how our methods work outside of our chosen parameter values for the generative models.

Our node-saving method was based on saving each node about to be deleted with some probability p_s, resulting in the preservation of nodes in "outer core". We would like to further investigate methods that require nodes to be saved based on different conditions. One could check the centrality values of the nodes to be removed at each iteration and save a proportion of those that have higher centrality values with some higher probability. Preserving these nodes with a higher probability may give us a sense of nodes that do possess "importance" but are removed in the k-core method, integrating the two notions of importance, being in a higher k-core and having higher centrality values.

To further investigate methods that may give us a larger magnitude increase in key centrality metrics between the original k-core decomposition and our methods, we could modify our edge-adding method on the basis of rewiring edges. This would involve identifying the edges that are to be removed at each step of the decomposition. This edge would be between a node pair involving a node in the k^{th} core and a node outside the k^{th} core. Removing this edge and the node outside the k^{th} core, one could add another edge, uniformly at random, with probability p, between the identified node from the node pair that resides in the k^{th} core and another node inside the k^{th} core. This process would then be repeated for each step of the decomposition. One can then check whether this will result in differences of larger magnitude of degree and betweenness centrality of the shells generated by the modified k-core method and the shells generated from the original method.

Conclusions. The ideas proposed in Barberá et al. (4) and Kitsak et al. (3) suggested that the most influential spreaders of a network reside in the higher k-shells, which motivated us to study the more influential nodes of these higher k-shells of networks. However, due to the limitations of the k-core method, we saw that most of the nodes are very commonly placed in the highest k-core, particularly for larger networks, making it hard to distinguish between the more influential nodes in that k-core. Thus, we were motivated to study the structure of networks and where influential nodes lie, specifically investigating a way to modify the original k-core methods to distribute nodes across shells in a more meaningful manner, to be able to better differentiate between the central nodes. We developed two modified k-core methods that involved either saving nodes being removed with some probability p_s, or adding some portion p_a of edges not present, at each iteration of the k-core decomposition. Our results showed that when our node-saving methods were applied to networks with a power-law degree distribution, such as the Holme-Kim and Caltech Facebook network, their nodes were spread out into more shells than when the original k-core method was used. For the edge-adding method, a similar trend of increased spread of nodes in the higher k-shells was seen when it was applied to the Barabasi-Albert model and the Caltech Facebook network. For both modified methods, there was a statistically significant increase in the degree, eigenvector and betweenness centrality measures for highest k-shell generated by the modified method, in comparison to those generated by the original k-core algorithm. For the node-saving method, this was seen in the Holme-Kim model and the Caltech Facebook network and for the edge-adding method, this was the most prominent in the Barabasi-Albert model and the Caltech Facebook network. Since most empirical networks relatively follow a power-law degree distribution, our methods open the door to better understanding of the highest k-shells the more influential nodes reside in.

Materials and Methods

A template of the psuedocode for both node-saving and edge-adding k-core algorithms is listed as follows.

Node-saving Method. Follow original k-core procedure except delete each node with probability p_s, otherwise keep the node for that step.

Edge-adding Method. Our edge-adding k-core measure, with parameter p_a representing percentage of edges added, is adapted from the algorithms for original k-core method in the Brain Connectivity Toolbox (10).

1. Start the algorithm for 1-core as the original k-core methods by removing nodes of zero degrees within the network
2. Proceed to 2-core and remove nodes with degree $k < 2$
3. After a round of removal, add M edges into the remaining network. Here, M is equal to the specified percentage p_a of the number of possible edges between n remaining nodes in the network.

$$M = p_a \times (n \times (n-1)/2 - existingedges) \quad [1]$$

   Each of the M edges is added between one node, chosen uniformly at random from nodes in the remaining network, and another node chosen uniformly at random from nodes that do not previously have an edge with the first chosen node.
4. Sort through the remaining nodes and remove nodes of degree $k < 2$
5. Repeat steps 3 and 4 until there is no nodes of degree $k < 2$. Then, 2-core is obtained and proceed to the next shell.
6. Repeat steps 2-5 for each shell and stop after reaching the highest shell.

ACKNOWLEDGMENTS. We acknowledge and thank Prof. Mason Porter and Abigail Hickok for their invaluable guidance not only with this project but throughout the academic quarter.

References.

1. MEJ Newman, *Networks: an introduction*. (Oxford University Press, Oxford; New York), (2010).
2. P Csermely, A London, LY Wu, B Uzzi, Structure and dynamics of core/periphery networks. *J. Complex Networks* **1**, 93–123 (2013).
3. M Kitsak, et al., Identification of influential spreaders in complex networks. *Nat. Phys.* **6**, 888–893 (2010).
4. P Barberá, et al., The critical periphery in the growth of social protests. *PLOS ONE* **10** (2015).
5. AL Traud, PJ Mucha, MA Porter, Social structure of facebook networks. *Phys. A: Stat. Mech. its Appl.* **391**, 4165 – 4180 (2012).
6. R Albert, AL Barabási, Statistical mechanics of complex networks. *Rev. Mod. Phys.* **74**, 47–97 (2002).
7. DJ Watts, SH Strogatz, Collective dynamics of 'small-world' networks. *Nature* **393**, 440–442 (1998).
8. P Holme, BJ Kim, Growing scale-free networks with tunable clustering. *Phys. review. E, Stat. nonlinear, soft matter physics* **65**, 026107 (2002).
9. P.Erdős, A.Rényi, *On Random Graphs I.* (1958).
10. Brain connectivity toolbox (2020).
11. AA Hagberg, DA Schult, PJ Swart, Exploring network structure, dynamics, and function using networkx in *Proceedings of the 7th Python in Science Conference*, eds. G Varoquaux, T Vaught, J Millman. (Pasadena, CA USA), pp. 11 – 15 (2008).