Identifying Relevant Links in Complex Networks

Jurjan Theodorus Ulbe Wouda Kuipers

MSc Emerging Digital Technologies
Department of Computer Science
University College London

Academic year: 2022/2023
Supervisor: Giacomo Livan

Abstract

This paper investigates the different ways that the relevant links of complex networks can be identified, as well as proposing a way to compare the effectiveness of network filtering techniques using synthetically generated networks. Filtering out irrelevant links in complex networks is a technological problem which has existed since the early 2000s, but due to no definitive correct answer existing it is difficult to assess whether filters produce correct or proper results. By introducing the testing of filtering methods with a synthetic network, the factors which affect the calculated backbones of filters can be identified, to gain a better understanding where specific filtering techniques can be best applied.

First, literature research will be explained, outlining the main topics that are looked at in this paper, as well as giving a brief description of different types of filtering methods. Furthermore, how the building of synthetic networks works is demonstrated. The methodology used to analyse the effects of synthetic networks on filtering techniques is that the filters are first tried on a real-life network, after which they are applied to a uniformly generated network. In order to assess what parts of a network affect the results, the generated dataset will undergo three different alterations to its data, and it is investigated whether two filtering techniques can identify these new anomalies, as well as how this affects the final extracted backbone. Using the data from the real-life and synthetic network in conjunction allows for an analysis to take place of what features of a complex network change the results of different filtering techniques.

The data gathered in the results section of this paper allows for conclusions to be made that the two filtering techniques which are compared in depth react differently to changes made to the synthetic network. This indicates that, dependent on the network that filters are applied to, the outcomes could be better predicted based on certain features of a network. While this does not give a definitive answer to what filter functions best, it gives a greater insight into how these filters function, and in what situation they are most applicable. Using this methodology and the research done, the effectiveness of using synthetic networks to test filter's reactions to different types of data is demonstrated, showing the value in this proposed solution to identifying which filtering method is most effective in certain cases.

Contents

Chapter 1: **Introduction**

In our current technological climate, different types of data are constantly being gathered to allow researchers to obtain a better insight into how our surroundings function. Observations made can give people a more advanced understanding of how to use or improve upon an existing system. One of these types of data is information regarding complex networks. This datatype exists of numerous points and links between them, where there is a constant flow of information. Data such as this is gathered daily, in social media platforms, banking applications, and transportation networks.

A compelling feature of these data types to investigate, is how data is spread through such a network. This is essential to know, as it can provide information on what parts are most essential for the system to function effectively. This can be described as the process of extracting a network's backbone, and with that finding out which links are most relevant. There are a number of network features that can be looked at, as well as a number of techniques which allow for the calculation of such a network backbone. However, some challenges exist.

The study of complex networks is a relatively new area and has been researched since the early 2000s. Since then, many methods have been developed in order to identify the most significant links. One issue is however that due to the nature of the results, it is hard to conclude whether the outcomes of backbone calculations provide a "correct" answer. This is due to all methods producing slightly different results, dependent on what factors are used for calculations. The current solution is simply to compare outcomes to a specific filter, which has been decided to be the benchmark to test against.
(Barabási and Pósfai, 2017)

An alternate solution which could be implemented is first testing out filtering methods on a synthetic network, to see whether it is fit for application on specific real-life networks. This concept would take into account that not only real-life networks are very different from one another, but also that different filtering techniques have different strengths in identifying certain network features for extracting a backbone. By first testing the filters on synthetic networks, it can be investigated what filters identify better than others. Based on these findings, the appropriate filter can be applied. This produces an outcome where using “ground truth” testing with a synthetic network, the ideal network filtering techniques can be identified for backbone extraction on networks. In practice, this would be done by testing filters on a synthetic network and then altering the network to identify changes in the outcome.

This process can provide a unique insight into how backbone extraction methods function, and which method would be most appropriate to use in specific cases. This overcomes some of the challenges of comparing filters, as instead of finding one right answer when it comes to calculating relevant links, this method identifies where different filters excel compared to others. This paper will investigate this using two different filtering techniques, to explore how synthetically generated networks can be used to better understand information from network filtering methods on real-life networks.

Chapter 2: **Literature Review**

2.1 Basic Network Features and Definitions

Network data is a type of data representation in which the connections between nodes are stored in a large table. These connections between data points are called edges, and have certain weights linked to them. They are able to portray complex systems which would otherwise be difficult to describe and do various calculations with. There are a number of ways in which networks can differ from another, firstly by being either directed or undirected. This means that in a directed graph edges specifically go from one node to another in a given direction, while in an undirected network the direction is interchangeable, with information flow between nodes being possible in both ways. One example of directed network data is banking data, where bank accounts can be represented as the nodes, the edges the directed payments between accounts, and the weight being the amount sent within transactions.

The weight of nodes is also an interchangeable feature for complex networks. There can be both weighted and unweighted networks, meaning that the weight of edges is not necessary for basic computations to take place. One example where no edge weights are required is in a network illustrating connections between people on a social media platform. In this case, nodes are users, and edges represent friend connections between them, where weight is not necessary as these links either exist or they don't.
(Bettilyon, 2019)

With any given network dataset, there are numerous relevant features that can be calculated to describe in more complex detail how a network is set up. First, the order of a graph is defined as the number of nodes in a network, while the size is the number of edges present. Each node also has a degree, which for an undirected graph is the total number of edges the node has, while for a directed graph this is split into the in- and out-degree. These values can also be used to calculate the average degree of a network, as well as using these features to find the average path length between nodes, which is the number of links needed to go from one randomly selected node to another. Lastly, a network's density is the ratio of edges present against the total number of possible edges, and the graph diameter is the largest path length present.

There are also several features which relate to a network's overall connectedness. The first is the number of connected components. A connected component is defined as a subgraph or group of nodes where at least one connection is present. A graph sometimes consists of multiple components, where some groups have no connections to others. The connectedness of individual nodes can also be defined through centrality measures, which provide a numerical value for the importance of nodes for information flow in the network. Two types of centrality measures are the closeness and betweenness centrality, where the first looks only at the average distance of links to other nodes, while the betweenness centrality quantifies how many shortest paths pass through a given node.
(Embl-Ebi, 2023) (Neo4j Graph Data Platform, 2023)

Using the features above, numerous elements for a network can be calculated, such as identifying major hubs present, and using this information to answer real life questions, such as what would happen if a network would experience a major disruption between nodes. These features will be used in this paper to analyse and compare different types of networks to one another.

2.2 Network Backbones and Filtering

A feature which is more difficult to calculate for a given network is identifying a network's most significant backbone. This is defined as the filtering out of weaker relations and redundant edges, to generate a sparser graph which still retains the initial network's most relevant links and significant properties. This means removing as much unneeded data as possible while having the network still display its overall connectedness and basic structure and retaining the most important flow of information throughout. This is a challenging feature to calculate, as a simpler graph needs to be generated, while maintaining most of those global properties. This new graph is then the network's most significant backbone.

Two filtering techniques which can be used for this are the hypergeometric cumulative distribution filter, and the Polya Urn filtering method. Both these methods apply different principles to identify a network's underlying backbone and produce distinctive results.

2.3 Hypergeometric Cumulative Distribution Filter

One of the main filtering techniques which will be used to derive backbones of network data within this paper is the hypergeometric cumulative distribution filter. This filter is based on random sampling of data without replacing it in groups. It calculates the probability of an event happening a specific number of times after a chosen number of trials, when sampling from a population without replacement. This can be illustrated using the formula below:

$$p = F(x \mid \mathrm{M}, \mathrm{K}, \mathrm{N}) = \sum_{i=0}^{x} \frac{\binom{K}{i}\binom{M-K}{N-i}}{\binom{M}{N}}$$

Within the above equation, the size of the population is "M", the number of times a sample is drawn is "N", "x" is the probability of event occurrence one is testing for, and "K" is the number of samples with the desired characteristic within the population. In order for the calculation to work, "M" must be larger than "K", and "N" cannot be bigger than "M".
(MathWorks, 2023) (Frost, 2023)

When applying this formula to network data, these variables can be substituted for the relevant parts in a network. To apply this filter to test whether an edge is relevant to the network's backbone, each edge needs to be tested producing a probability value for each. The variables used are the following for a link from node A to B:

Link A->B

- **x** = the weight from A to B
- **M** = the out-strength of A
- **K** = the in-strength of B
- **N** = the sum of all strengths of all nodes in the network

The probability which is calculated using this method can then be used to validate the edges within the network. Null hypothesis testing can be done, and if a link obtains a probability below a certain alpha significance level, it can be validated to the network's backbone. This validation method works the same way as other network filtering techniques, and the validated backbone can thus be compared to results of other filters.

(Marcaccioli and Livan, 2019) (Encyclopædia Britannica, 2023)

2.4 Polya Urn Filtering Method

The other main filtering method that will be observed to validate links within network datasets is the Polya Urn Filtering method. This method is similar to the hypergeometric cumulative distribution filter, as the probability of an event occurring is calculated after taking a sample from a population a number of times in a row. The way in which it differs however, is that samples are added back to the population after each trial. This means that the number of samples in the population increases by one after each trial, which produces unique results using this method. This can be further explained using the formula below:

$$\mathrm{P}(\, w \mid \mathrm{k}, \mathrm{s}, \mathrm{a}\,) = \binom{s}{w} \frac{B(\frac{1}{a} + w, \frac{k-1}{a} + s - w)}{B(\frac{1}{a}, \frac{k-1}{a})}$$

Similar to other filtering techniques, the probability or p-value calculated is compared to an alpha significance level to validate whether links are a part of the network backbone. The Polya Urn method is further unique from the hypergeometric cumulative distribution filter, as it introduces an "a" parameter which can be tuned to alter the effectiveness of validation for network edges. It is used to affect how much the network's growth preferential attachment is taken into account when doing trials, as nodes are more likely to created edges with other nodes which already have more links.

These features make it an interesting filtering method to use in conjunction with the hypergeometric cumulative distribution filter, as differing results will provide a better insight into what links are relevant in networks with different features.

(Marcaccioli and Livan, 2019)

2.5 Other Filtering Methods

2.5.1 Disparity Filter

There are a number of other network filtering methods that could be applied in order to extract the backbone within a complex network which are not discussed in this paper. However, these methods provide a different solution to the problem. The first is the disparity filter, which is a technique that is often applied as the benchmark to compare the effectiveness and outcomes of other filters. This filter is similar to others as it works by discarding nodes which are below a necessary weight threshold and looks at the importance of nodes by comparing its total weight to the number of links present. A normalized weight is calculated for each node within the network, after which a p-value is produced. This calculated number is the probability if the normalized weight is larger or equal to a given alpha significance level. Using this information, the relevant backbone in a network dataset can be calculated.

(Serrano, Boguñá and Vespignani, 2009) (Marcaccioli and Livan, 2019)

2.5.2 Maximum Likelihood Filter

The maximum likelihood filter applies the same concepts used in maximum likelihood estimation, which takes data as inputs to estimate parameters using a given probability distribution. Similar to other filtering techniques, the inputs are the node in- and out-weights and weights of specific edges, and each edge is assigned a significance score based on where they fall on a marginal distribution. Using these derived significance values, edges can be validated as relevant to the network backbone by comparing it to an alpha significance level.

(Marcaccioli and Livan, 2019) (Dianati, 2016) (Brooks-Bartlett, 2018) (Frost, 2023)

2.5.3 Enhanced Configuration Model

This network filtering technique is a more technical improvement of the normal configuration model. This model is a method for generating a network when given a degree sequence dataset. In this case, instead of generating a random network, the enhanced configuration model instead uses the degree sequence information of a given dataset to reconstruct it, and in this way identifying the relevant backbone present.

A number of properties can be derived using this model and its inputs, including the probability of edges occurring between nodes, the degree distribution and the clustering coefficient. Using the calculated information, the model can infer the existence of numerous important features present, such as smaller and larger components appearing in the network. By therefore reconstructing the initial network from basic data, a backbone within that network can be derived.

(Marcaccioli and Livan, 2019) (Mastrandrea, 2014) (Newman, 2019) (GeeksForGeeks, 2022)

2.5.4 Noise-Corrected Filter

Lastly, the noise-corrected filter proposes a solution to filtering out noisy data within networks, by removing edges providing limited or unnecessary data to the total network. This method uses both the in- and out-weights of nodes and assumes that edge weights in the network are taken from a binomial distribution. Next, the error-variance for each edge is calculated, which is defined as how far each edge weight is from the average. This calculated value is used to validate whether edges are identified as noisy data, and thus can be filtered out of the network. Finally, this produces the significant backbone. While these methods provide a further insight in the workings of different filtering techniques, this paper will focus mainly on the hypergeometric cumulative distribution filter and the Polya Urn filter.

(Marcaccioli and Livan, 2019) (Study.com, 2023) (Coscia and Neffke, 2017)

2.6 Null Hypothesis and Bonferroni Correction

Using the above methods, almost all produce a probability value for each edge within a network at the end of calculations. This p-value is then compared to an alpha significance level to determine whether a link must be validated or not. One important element which needs to be introduced before validating edges is the Bonferroni correction, which adjusts the chosen alpha significance level in order to lower the risks of errors when validating. If an occurrence within a dataset is rare, but numerous tests are done, the probability of observing these rarer events is no longer accurately represented as it increases with the number of trials done. This means that when multiple null hypotheses are tested these must be rejected when increasing the number of tests. This can be accomplished using the following formula:

$$P = \frac{\alpha}{m}$$

This is the Bonferroni correction, where "α" is the initial alpha significance value, and "m" is the number of hypothesis tests done. This means that edges in a network should only be validated if their p-value is lower than the alpha value divided by the total number of edges present. This produces better results as only very significant links are added to the backbone in larger networks.

(Marcaccioli and Livan, 2019) (Armstrong, 2014)

2.7 Barabasi Albert Method for Building Networks

One aspect which will be looked at later in this paper is using synthetic network datasets in order to test how edges are validated, what affects the identification of a network backbone, and whether changes to a backbone can be detected.

In order to do this, first a synthetic network must be created which follows the basic structure of real-life networks. One method for this is using the Barabasi Albert model. This model takes a number of factors into account to generate networks of any scale. This is done largely by taking into account growth and preferential attachment within network models. When generating a network using other models, such as the Erdős-Rényi model, before any links are randomly created, a fixed number of nodes is set for the network. This means if it is decided to produce a network with 100 nodes, the model will only create links between these. This however defies how real-world networks behave, where if a new node is added to a network, this should continually generate new nodes, meaning that there should not be an initial fixed number.

This is one of the main assumptions that the Barabasi Albert model aims to mitigate, by accounting for this growth, and at every step adding more nodes continuously to reproduce real networks more accurately. Furthermore, it accounts for preferential attachment. This is defined as the more connected a node already is, the more likely it is to receive edges with other nodes. It therefore considers the number of adjacent nodes, in order to calculate the probability of a new edge forming.

Using the Barabasi Albert model, an accurate synthetic dataset can be produced so that it is most similar to real-life networks. This provides informative opportunities in network backbone testing, as many parameters of synthetic models can be altered to investigate how the most important edges in a network are identified using different filtering techniques. The formula for how this model calculates the probability of edges forming is displayed below:

$$PA(x,y) = |N(x)| * |N(y)|$$

(Barabási and Pósfai, 2017) (Neo4j Graph Data Science, 2023)

Lastly, the nodes and edges produced using this model are given strengths randomly from a power law distribution. This means that after the production of the list of links, the places where links are created are replaced by a randomly generated weight producing a usable adjacency matrix for backbone calculation. A power law distribution can be applied to produce different types of weight distributions, by tuning the tail exponent to either make more heterogeneous or homogeneous distributions. The random weights drawn from this are then used in further calculations. By tuning the tail exponent, more control is provided over what the network looks like, which can provide varying results when applying network filtering techniques.

Chapter 3: **Methodology**

3.1 Network Datasets used for Backbone Validation

3.1.1 Airport Network Dataset

In order to identify network backbones and to distinguish what factors affect the validation of edges within a large network dataset, two different types of datasets will be used. The first is the “Airport Network Dataset” from the Bureau of Transportation Statistics. This dataset contains data on over 250,000 flights taken in 2017 in the United States of America, containing various information on each. This includes origin and destination airports, carrier names and IDs, as well as the number of passengers.

Using this dataset, the elements that build up the network that will be used to identify significant backbones present. The nodes within the network are the airports, and the edges the flights between them. Furthermore, the edges are weighted using the passenger counts. When using this dataset some data pre-processing must be done initially. Firstly, this is done by removing repeat flights. As often edge weights are taken from the frequency of edges occurring, when using the passenger counts as weights this no longer works when multiple flights fly one route. To work around this, flights that occur between airports multiple times are merged into one link where their total passenger counts are added together for the edge weight. Furthermore, any edge with zero passengers is also removed from the dataset. A visual representation of the network is shown as follows, where the size and colour of nodes illustrate their degree.

(Bureau of Transport Statistics, 2017)

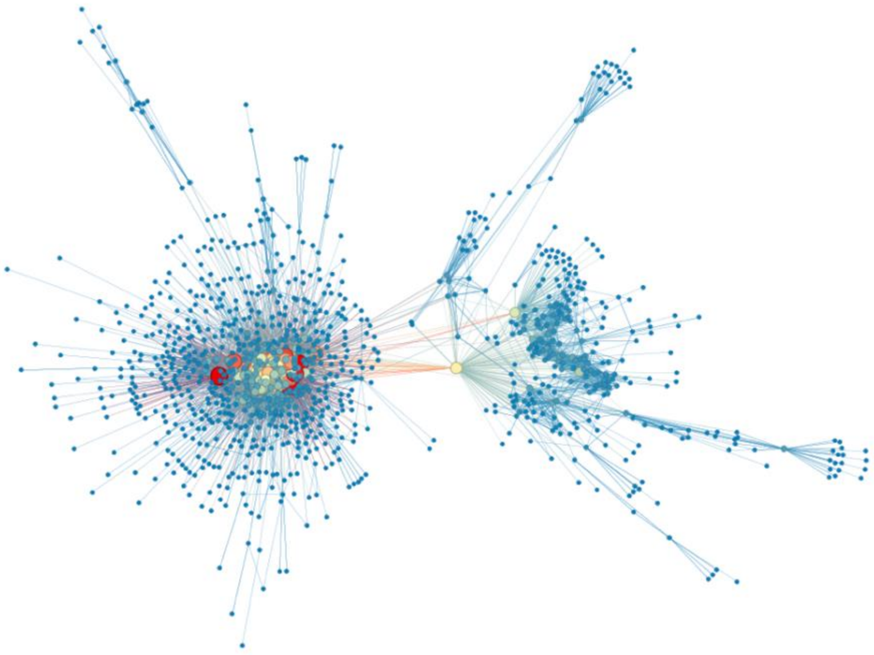

Figure 1: Visual representation of the Airport Network

3.1.2 Synthetic Dataset

To further investigate how backbones can be extracted using different methods, this paper will additionally be applying network filtering methods on a synthetic dataset. This network will be generated using the Barabasi Albert Method to create a dataset most similar to real-life examples. The benefit of using the real airport network dataset and a synthetic dataset in conjunction is that they allow for different observations to be made, as the generated network allows for more control of the number of nodes and edges which are created, while the airport network provides information about how the filters work in a real-life example. This can give an alternative insight into the factors affecting the backbone extraction, and comparing the results from these two network types produces a better understanding of what changes to a network affect the filtered outcome.

The way in which this is implemented in the MATLAB code is by using a number of set variables, including the desired number of nodes, the number of nodes in the initial core, and finally the number of links formed by new nodes. An adjacency matrix is then created for the initial core where all nodes are fully connected. Next, a loop iterates through the desired number of nodes, obtaining the degree sequence, and probability of a node linking with other nodes, after which new nodes and links are added to the adjacency matrix. The final step in producing the network is setting the edge weights. Within the code, two different power law distributions are made with differing tail exponents. The created adjacency matrix is then iterated through and draws weights from the power law distributions at random, rounding them to the nearest integer. This leaves two identical networks with differing weights which can be tested for their network backbone. These distributions are derived from normal distributions, but with differing standard deviations. This results in the two distributions shown below.

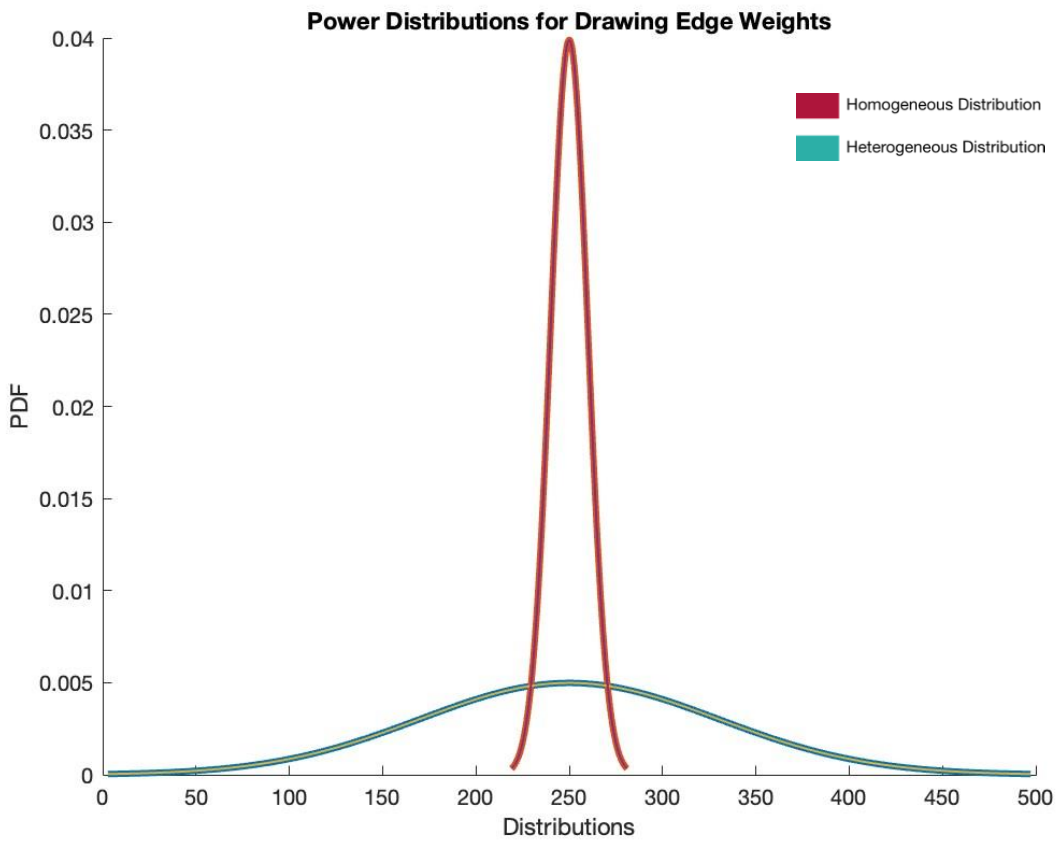

Figure 2: Power law distributions for drawing edge weights with different tail exponents

Two different network filtering methods will be applied to these two datasets in order to obtain more varied results. The methods used in this paper are the hypergeometric cumulative distribution filter, and the Polya Urn method.

3.2 Altering the Synthetic Dataset

3.2.1 Ground Truth

The idea of ground truth in statistics is derived from the process of training machine learning algorithms with a dataset where the accuracy of the algorithm can be tested against known, real world examples. When machine learning algorithms are trained, two different datasets are used, training data and testing data. First, training data is applied to educate models to learn to make appropriate decisions, after which the effectiveness of this training can be evaluated using the testing data. This works, as with the testing data the desired outcome is already known, so that one can determine whether the model is reproducing this correct result.

While the use of a ground truth dataset is mostly exclusively used in the field of training machine learning algorithms, the same fundamentals can be applied to the observations made in this paper.

(Technopedia, 1970) (MathWorks, 2023) (Manager and Zabolotny, 2023)

A challenge which the majority of network filtering methods face, is whether the results from simplifying networks to their significant backbone can be viewed as correct. This is evident, as on the same dataset, different filtering methods produce vastly distinctive results regarding which edges are validated or not. This is due to two main factors, including the fundamental calculations which are done to produce p-values for links, as well as what alpha significance is used to validate whether calculated p-values are validated or not. To then subsequently evaluate if these results are effective in filtering out irrelevant data, they are compared to the results of the Disparity Filtering Technique. This is due to this method being one of the most widely used ways of filtering, and therefore is applied as a benchmark for others. One could however further attempt to determine a method's effectiveness by using filtering techniques on a dataset with "ground truth", in this case a synthetically generated network. This gives more insights into how different filtering techniques validate links, allowing for a better understanding of what can be seen as the "ground truth" when extracting a network's backbone.

(Marcaccioli and Livan, 2019)

3.2.2 Synthetic Network Alterations

The advantage of employing a synthetically generated network is that it allows for more control over what the data looks like, as well as deciding what the ground truth is when finding the network backbone. Furthermore, this generated network can be altered to be able to make more observations into how backbones can be extracted, as well as if certain network filtering techniques are able to detect anomalies and changes to networks. This paper proposes a way to alter synthetic networks in three different ways, to analyse the changes when identifying the relevant links present.

3.2.2.1 Creating Additional Edges

The first change made after the creation of a synthetic network is adding additional edges between existing nodes. This works by selecting specific nodes from the dataset and giving these a random number of new edges. Hypothetically, this should change what edges are validated, as this results in more major hubs appearing throughout the network. This change will be tested for using the initial network and comparing it to the network after these alterations have been made.

New edges are created using a number of variables. The first being the probability that a node is selected from the nodes list. If selected, a different probability value is used to allow this node to create links to other nodes with a success rate dependent on the chosen probability. Finally, it is important to note that old links are not altered, and that new links get assigned weights drawn from the same power distribution used to generate the initial network.

3.2.2.2 Changing Weights of Individual Links

The second option when altering the synthetically generated network is to change the weights of already existing links in the dataset. This can be done similar to the first alteration, where a loop iterates through every node, and using a set probability will choose whether nodes are affected. If a node is selected at random, its already existing links to other nodes get slightly decreased or increased using a gamma value. This means that every one of its weights are changed by a multiple of 1 plus the chosen gamma value as shown by the formula below:

$$w_{ij} = w_{ij}(1 + \gamma)$$

This is done to both the in- and out-going edges of the node, after which the new value is rounded to the nearest integer. This new data should provide differing results when compared to the initially generated network, as specific nodes will have a more important role in connectivity due to greater weights, meaning that these newly calculated edges may be included in the network backbone.

3.2.2.3 Reshuffling Link Weights at Random

The final change which will be made to the synthetic network dataset is reshuffling the weights of existing links at random. This simply means that all edges which have been generated will get their weights replaced by another edge, so that the same edges do remain but with different weights. This will result in a very different network, however containing the same total network weight present.

Within the MATLAB code this is achieved by producing an array of all the edge weights from the adjacency matrix. The order of this array is then randomized, after which the links get assigned a new weight from the array. This produces the same network but with shuffled weights.

3.3 Edge Validation with the Hypergeometric Cumulative Distribution Filter

In order to calculate backbones present in the two types of networks above, the airport network and synthetic network, two different filtering techniques will be applied. The first is the hypergeometric cumulative distribution filter. As stated earlier, this method works by sampling data randomly without replacing them after multiple trials in populations. A probability can then be calculated for the chance of sampling a certain number of desired outcomes in a specific population. This probability, or p-value, is calculated for each link present in the network's adjacency matrix and is then used to validate whether links should be included in the network's backbone.

To produce these p-values for each link present, a MATLAB function has been created which can be ran multiple times to calculate the probability value for every edge in an input adjacency matrix. This function takes as input firstly the edgelist that it should be applied to. Next, a number of important features are derived from the matrix, such as the size, the strengths of links, and the total number of edges present. The number of edges present is important, as this is used for the Bonferroni Correction of the alpha significance value to which p-values will be compared to for validation. This is then done by dividing a set significance value by this derived variable, which produces a new value in order to minimize errors and allowing only truly significant edges to be validated.

Lastly, three values are calculated which are used for the hypergeometric filter calculation. First, this is the total strength of the network, which simply produces the sum of all edges in the adjacency matrix. Next, the in-strengths and out-strengths of nodes are calculated, by obtaining the sum of the rows and the sum of columns of the matrix respectively. These lists of strengths are then stored as single row or column matrices whose values can be accessed using the relevant index.

Finally, the actual filtering calculation can be applied. First, a new matrix is created using the matrix that was initially used as an argument for the filtering function where the calculated p-values will be stored. Next, the function iterates through the entirety of the adjacency matrix using a simple nested for loop, iterating through all rows and columns. Once an element in the matrix is located which contains a link, the variables used in the calculation are derived from the matrices from earlier.

For the first variable "x", the index of the edge in the matrix is simply used to get the weight of the edge. For "M", the previously created single column in-strengths matrix gets the in-strength of the node, and the same is done for "K" which obtains the out-strength. Lastly, "N" uses the sum of all strengths which is the total network strength. These variables are then input in the MATLAB "hygecdf" function, which calculates the p-value for a given link. With this function the argument "upper" is also set to true, as this returns the complement of the calculated probability at value "x". These values are pushed to the matrix containing results, after which these can be compared to the alpha significance level to see whether they should be validated. Using these steps, the backbone of any input matrix can be obtained using the hypergeometric cumulative distribution filtering method.

3.4 Edge Validation with the Polya Urn Filtering Method

The second filtering technique that will be used is the Polya Urn method. As described previously, this technique calculates the probability of something occurring after taking samples from a population numerous times. Instead of removing these samples from the population, they are added back after each trial. This is where this technique stands out from the hypergeometric cumulative distribution method, and can therefore produce informative and different results. The probability calculated for each link in the network is the p-value which is compared to the chosen alpha significance value, after which edges can be validated and added to the network's backbone.

The MATLAB code which is used to calculate the p-value for each of these links is taken directly from a paper which first implemented this method to calculate network backbones. However, some slight changes have been made to have the code work with the airport network and synthetic network for this paper.
(Marcaccioli and Livan, 2019)

The MATLAB code itself is built around the same principle as the code for the hypergeometric cumulative distribution function. This is due to the formula for calculating a p-value using the Polya Urn method being converted to MATLAB code, and then using a nested for loop to iterate through every known edge in the network's adjacency matrix. The main function which is used is the "PF" function. This function takes 4 input arguments to produce the p-values. Firstly, it takes the adjacency matrix, which is a matrix of all nodes in the network, where the values within the matrix indicate the presence of links between nodes and their respective weights. Next it takes the free parameter of the filter. This free parameter "a" as previously described is used in order to account for the network's growth preferential attachment when sampling from the population. This parameter is useful as it can be tuned uniquely for each network that the method is used on, allowing for more flexibility when calculating the network backbone. When the value of “a” is smaller than 1, more elements of the original network are retained. If the value is set to 1, the filtering technique coincides directly with the Disparity filter. Finally, if the value is set greater than 1, more edges get filtered out. As the main changes to networks in this paper are done by altering the synthetically generated network, the value of “a” will be kept at a constant number.

The next input argument is the “apr_lvl” or the approximate form of the p-value which is used in the calculations. This approximation is simply used to speed up the computation of the p-values, and for the purpose of this paper will be set to 10, which triggers the use of this approximate form on every edge in the adjacency matrix. Finally, the last argument indicates whether parallel computing is used or not.

After taking these inputs, the function calculates similar features for the final Polya Urn calculation as the hypergeometric cumulative distribution function, including the total network strength, the in-strength and the out-strength for each node. Furthermore, the code checks for symmetry in the adjacency matrix, so that it can be used for both directed and undirected networks. Depending on the value of “a” input as an argument, machine learning estimates are lastly calculated. Finally, the p-values for each link are produced using the obtained network features. The values are similarly compared to the alpha significance level, which was corrected for errors using the Bonferroni correction, so that edges can be validated, and the network backbone can be extracted accordingly. Using this method in conjunction with the hypergeometric cumulative distribution method, varying results can be computed for each network, providing more information of how network backbones can be extracted.
(Marcaccioli and Livan, 2019) (MathWorks, 2023)

3.5 Producing Usable Data with Datasets and Filtering Methods

In order to create results which can by analysed later, the two filtering methods are first simply tested on the airport network dataset. This creates an understanding of how these methods function, as well as showing the different results produced when applied to a real-life example. Next, the filtering methods will be applied to the synthetic dataset, and the synthetic dataset after alterations to it have been made. In this stage the random element of altering the network is introduced, so multiple tests need to be done to see whether results have high precision. To do this, for each way that the network is altered, 100 synthetic networks changes are made, and tested for a backbone with both filtering methods before and after changes are applied. The average results of these changes can then be used appropriately to analyse whether these filtering methods can detect anomalies in the data.

Chapter 4: **Results**

4.1 Backbone Extraction on the Airport Network

4.1.1 Airport Network Statistics

Before any backbone extraction is done, some basic statistical features of the network have been calculated to get a better understanding of how the network is built. The dataset contains flights from 2017 taken in the USA and is made up of three main connected components. The average degree of a node is 17.9, meaning an average of 18 flights left and flew into each airport in 2017. Furthermore, the network diameter, or the shortest distance between the two furthest nodes, is 9 links. Lastly, the average path length between nodes is 3.40 and the average clustering coefficient is 0.439. This information can later be used to identify major hubs within the network, which are useful in the analysis of what elements of the network remain after backbone extraction, as well as comparing it to new network statistics.

(Baeldung, 2022)

4.1.2 Obtaining Initial Backbones

The first analysis that will be done is applying the two backbone extraction methods on the Airport Network dataset. This can first provide information as two how these techniques differ, but also allows us to get a better insight into what flights and airports are most significant in the network. In these first tests, the hypergeometric filter runs normally, while the arguments of the Polya Urn method have been set to “a=0.6” and “apr_lvl=10”. This results in the validation of edges illustrated below.

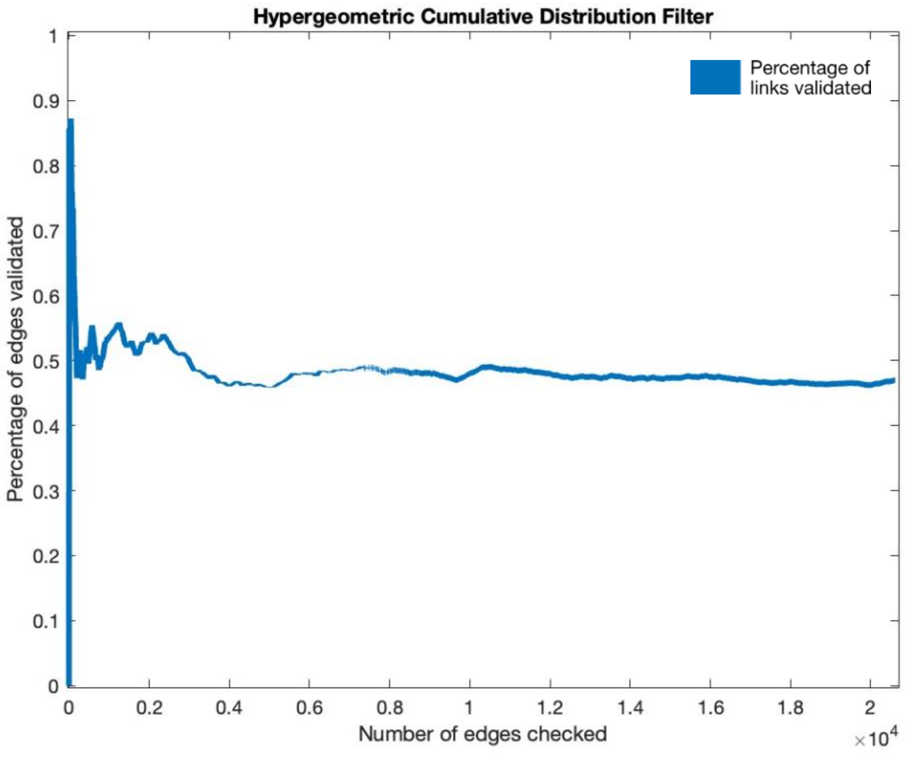

Figure 3: Hypergeometric Cumulative Distribution Filter applied on Airport Network

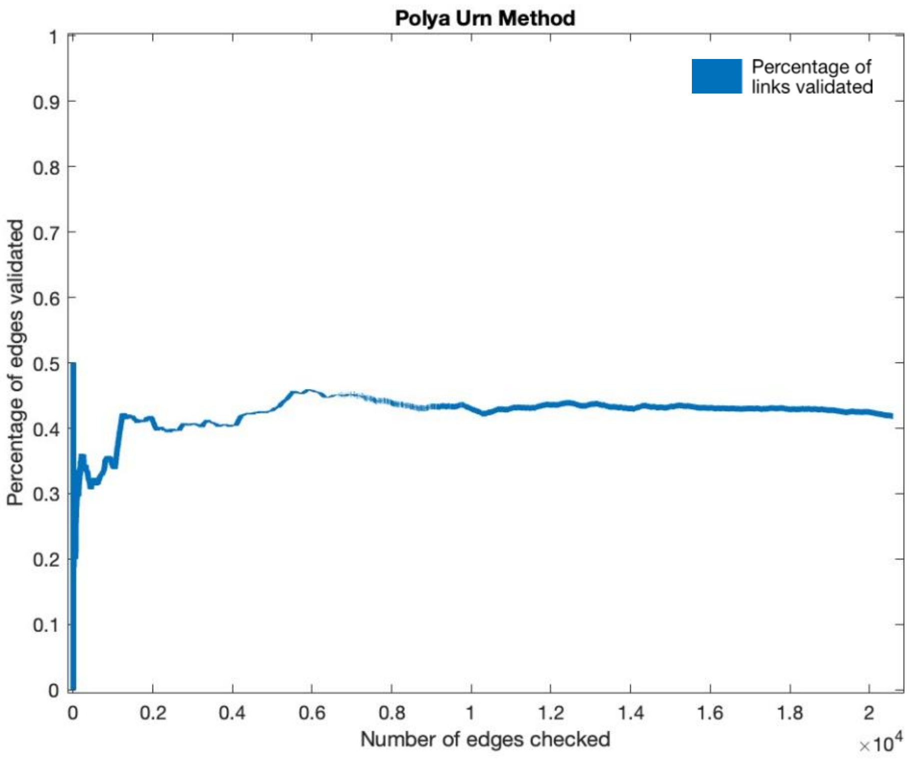

Figure 4: Polya Urn Method where "a=0.6" applied on Airport Network

In the plots above, the number of edges in the adjacency matrix that were tested for their p-value is plotted against the percentage of edges that were then successfully validated and added to the airport network's backbone. For the hypergeometric cumulative distribution filter, this resulted in the validation of 9689 edges, and for the Polya Urn method 8614 edges were validated and added to the backbone. This results in a percentage of 47.0% and 41.8% of validated links for each method respectively. Initially in the calculations, due to the small number of edges checked, the percentage still fluctuates, until it slowly settles. This is better illustrated in the two graphs below, which show the results for the first 1000 edges and 250 edges for the hypergeometric filter and the Polya Urn filter respectively.

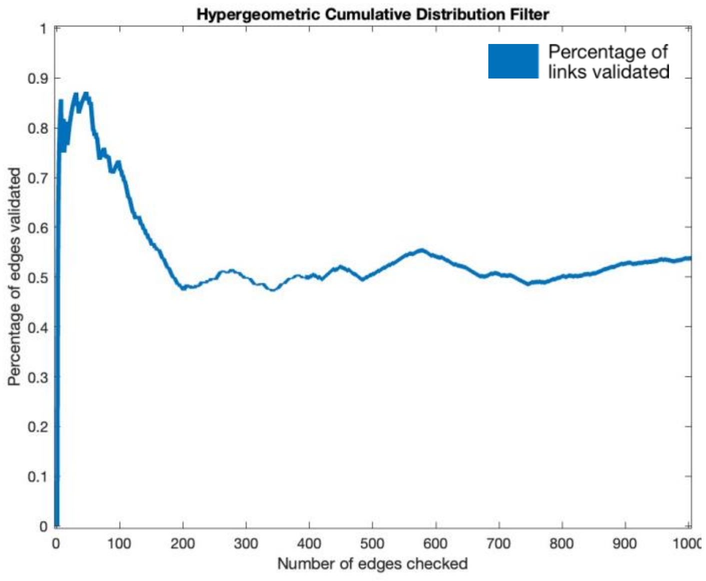

Figure 5: First 1000 edges checked using the Hypergeometric Cumulative Distribution Filter

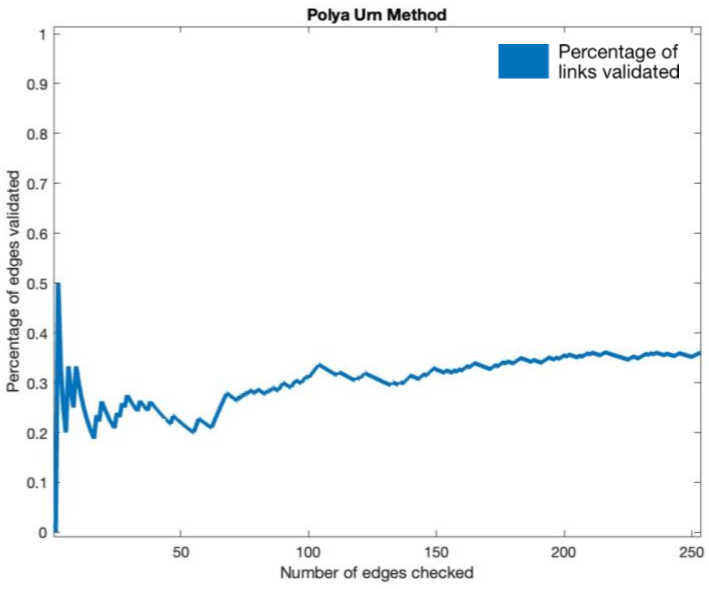

Figure 6: First 250 edges checked using the Polya Urn filter

Lasty, it can be noted that through tuning the "a" parameter of the Polya Filter, the two methods produce more contrasting results. This is due to when increasing the "a" parameter it increases the effect of taking into account the network's growth preferential treatment, resulting in less validated links. The same plot is shown below with "a" set to 1, which corresponds to the Disparity Filter. This results in 3712 edges added to the backbone, for a percentage of 18.0%.

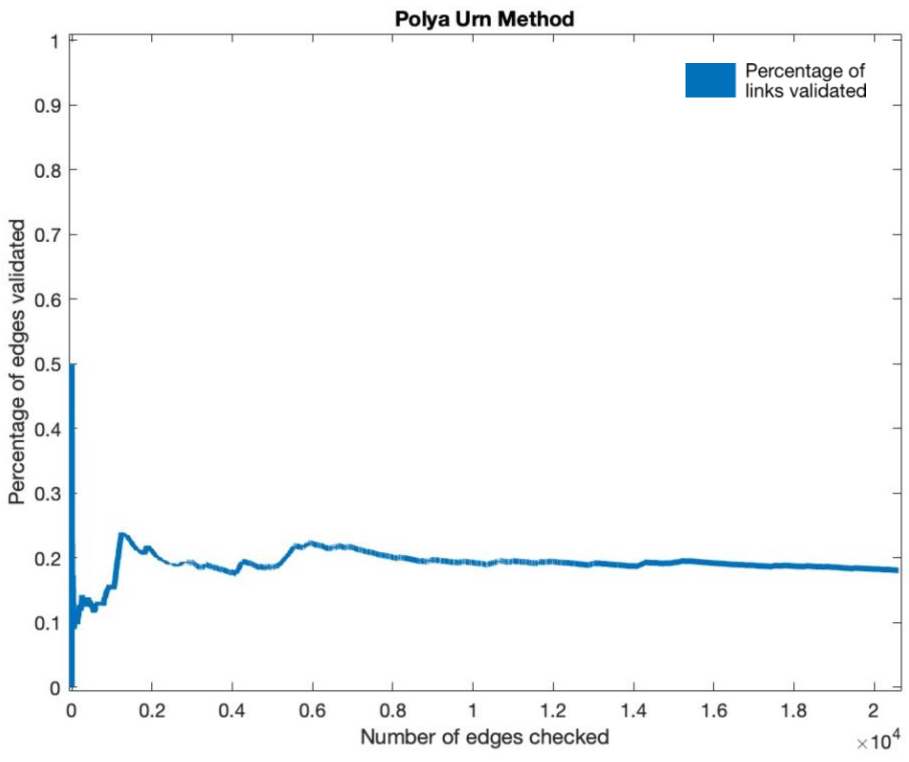

Figure 7: Polya Urn Method where "a=1" applied on Airport Network

4.1.3 *Observing Significance of Backbones on Major Hubs*

Within the Airport Network dataset, some nodes can be qualified as major hubs, due to the network's features such as the average degree, centrality, and clustering coefficients. To further investigate how the two methods have handled filtering out insignificant information, these hubs can be looked at to see if connections between them have been filtered out through the process. Three major hubs and that have been identified after looking at in and out degrees, their closeness centralities and betweenness centralities, are Dallas/Fort Worth (DFW), Los Angeles (LAX), and Atlanta (ATL). The calculated network statistics are shown below.

(Guen Song et al, 2017)

Table 1: Statistical features of major hubs in Airport Network

Airport	Degree	Closeness Centrality	Betweenness Centrality
Dallas/Fort Worth	389	0.449	57063
Los Angeles	348	0.453	43490
Atlanta	392	0.458	40582

Before filtering out links, Dallas has 389 unique flights containing passengers leaving and entering its airport in the year 2017. When first looking at the effects of the hypergeometric filter, one can observe that this value has decreased with only 264 of these flights remaining on the backbone. Some interesting edges to look at are the connections to other central hubs, where it is apparent that some of these have also been removed. For example, in 2017 over 870,000 passengers flew from Dallas Fort Worth to Atlanta, however due to the calculated p-value and validation, this link has been pruned from the network therefore not showing up in the extracted backbone. Similarly, the flights between Dallas and Los Angeles have been removed. Furthermore, the other network statistics used to identify the major hubs can be recalculated. After filtering out irrelevant nodes, Dallas now has a closeness centrality of 54275 and a betweenness centrality of 68.

The same can be observed for the other two major hubs, where the degree of Los Angeles has reduced from 348 to 112 links, and Atlanta declining from 392 to 249 links. Similarly, some major connections between these airports have been excluded, including flights between Charlotte and Atlanta, which normally contained 355,000 passengers. However, regular flights between Los Angeles and San Francisco, and Los Angeles and New York have remained on the backbone. Similar with flights between the other major hubs and Atlanta, some have been removed while others were necessary elements for spreading information in the network after filtering.

The same observations on major hubs can be made after extracting the backbone using the Polya Urn method. When first looking at Dallas, its total degree has reduced from 389 to 325. When looking at the connections to the other identified central hubs, links between all these airports remained after extracting the backbone. Furthermore, when looking at the 20 flights to and from these hubs containing the most passengers, it appears that these all remain on the backbone after filtering using the Polya Urn method. This is interesting, as even though the two methods removed a similar number of links, evidently the hypergeometric cumulative distribution filter omitted more links between major hubs. However, the Polya Urn method similarly sees lower degree numbers for the two other central hubs, Los Angeles reducing from 348 to 226, and Atlanta from 392 to 343. From these numbers it is concluded that the Polya Urn method retains more information regarding a network's central hubs as opposed to the hypergeometric filter. Below are the complete changed network statistics for both methods after backbone extraction.

Table 2: Comparing network statistics for the Airport Network before and after backbone extraction using the two filtering techniques

Filtering Technique	Degree	Closeness Centrality	Betweenness Centrality
Unfiltered	DFW: *389* LAX: *348* ATL: *192*	DFW: 0.449 LAX: 0.453 ATL: 0.458	DFW: 57063 LAX: 43490 ATL: *40582*
Polya Urn Method	DFW: *325* LAX: *226* ATL: *343*	DFW: 30214 LAX: 35398 ATL: 24441	DFW: *38* LAX: *38* ATL: *38*
Hypergeometric Cumulative Distribution Filter	DFW: *264* LAX: *112* ATL: 249	DFW: 54275 LAX: 27637 ATL: 38757	DFW: *68* LAX: *68* ATL: *68*

4.1.4 Recalculating Network Statistics after Backbone Extraction

Lastly, for the two backbones that were extracted using the hypergeometric cumulative distribution filter and the Polya Urn method, new network statistics can be calculated. This can provide an insight into how the connectedness and ability for information flow has changed and allows for comparison between the two methods. The table below illustrates these changes.

Table 3: Comparing average network statistics for the Airport Network before and after backbone extraction using the two filtering techniques

Network Statistic	Original Network	Hypergeometric Cumulative Distribution Filter	Polya Urn Method
Average Degree	17.878	8.544	11.214
Network Diameter	9	10	8
Graph Density	0.016	0.008	0.015
Connected Components	3	4	9
Average Clustering Coefficient	0.439	0.297	0.305
Average Path Length	3.397	4.038	3.296

The first large change that can be observed from the table above is the decrease in the average degree. This is an obvious difference, due to the filters removing a large number of edges from the network. What is noticeable however is the distinct change between the two filtering techniques. Above it can be seen that the Polya Urn Method retains a higher average degree per node in the network. This can again indicate that it has kept nodes with more connections, while pruning nodes and edges which had less links to other nodes. This is not the case for the hypergeometric cumulative distribution filter, which has a much smaller average degree, meaning that it has most likely validated smaller airports with less links to the extracted backbone. While this is the case, it has no large effect on the network diameter which remains similar throughout the three networks.

Next, when observing the graph density, there is a clear change between the two filtering techniques. The graph density of a network is defined as the proportion of edges present compared to the maximum number of possible edges. If the graph density were 1, that would indicate that all nodes in the network are connected to one another with a link. The initial density is 0.016, which remains mostly unchanged for the Polya Urn filter, where it lowers to 0.015. For the hypergeometric cumulative distribution filter however, the density has lowered to 0.008. From this one can assume that while the network diameter remains the same, the hypergeometric cumulative distribution filter retains more of the initial nodes but uses less connections between them to produce its backbone.

(Datta, 2022)

Another interesting observation is the reduction in the number of connected components present. In this case, the results from the hypergeometric cumulative distribution filter stay similar to the initial network, where the number of components has increased from 3 to 4. After backbone extraction using the Polya Urn filter, the network is now made up of 9 connected components. This means that after extracting the backbone using the Polya Urn filter, the network has split into 9 subgraphs in which nodes can only connect with one another.

(Baeldung, 2022)

The next statistic demonstrates how nodes cluster together in the network. The clustering coefficient of a node is defined as a measure of how the neighbouring nodes connect with one another. In the initial airport network, the average clustering coefficient is 0.439, however after filtering this decreases for both methods, to 0.297 and 0.305. This indicates that there are less clusters after extracting the backbone, which is to be expected. Although, this does not mean that nodes are less well connected, but rather that unnecessary links between neighbouring nodes have been omitted in the backbone.

(Newman, 2006) (GeeksForGeeks, 2022) (Bettilyon, 2019)

Finally, the average path length is a measure which gives the average number of connections needed to travel between two nodes in the network. While the hypergeometric cumulative distribution filter exhibits a slight increase, the Polya Urn method has a small decrease in average path length as compared to the original network. This shows that although many existing links have been removed to calculate the backbone, the number of connections needed to relay information has not changed massively, meaning that the filtering techniques have successfully removed irrelevant links while retaining those that have a significant use in the network.

4.2 Backbone Extraction on Initial Synthetic Network

4.2.1 Generating a Synthetic Network

4.2.1.1 Initial Network and Network Statistics

The initial generated network that is being used for trials with the two filtering techniques is made using the Barabasi Albert Model, with 6 initial nodes in the core, 1006 nodes overall and each new node making 4 links. The edge weights are then drawn from a power law distribution at random. This produces a network that can be visually represented as follows, where the size and colour of nodes represent their degree.

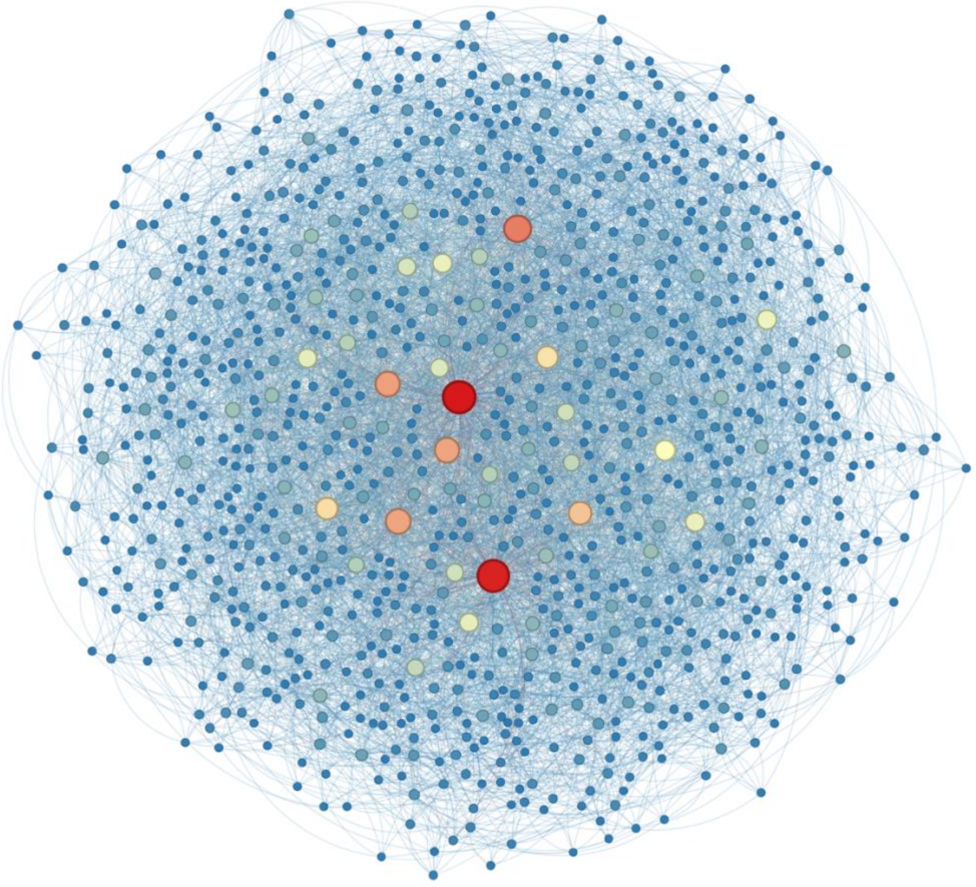

Figure 8: Synthetically generated network with 1006 nodes using the Barabasi Albert Model

The type of network shown above is the same as which will be used for network alterations. Compared to the airport network it is more uniform, where a clear core and other elements can be identified. To demonstrate the expected features of such a model before any changes have been made, the basic network statistics are shown below.

Table 4: Basic statistical features of a synthetically generated network

Network Statistic	Synthetic Network
Nodes	1006
Edges	7942
Average Degree	7.895
Network Diameter	5
Graph Density	0.008
Connected Components	1
Average Clustering Coefficient	0.032
Average Path Length	3.199

4.2.2 Filtering Techniques on the Initial Network

To demonstrate the different effects that backbone extraction has on such a network, both techniques have been applied. When first applying the hypergeometric cumulative distribution filter and the Polya Urn filter in the same way as had been done to the airport network, they provide contradictory results. With the “a” parameter of the Polya Urn filter set to 0.6 as was done previously, links get validated in the following way.

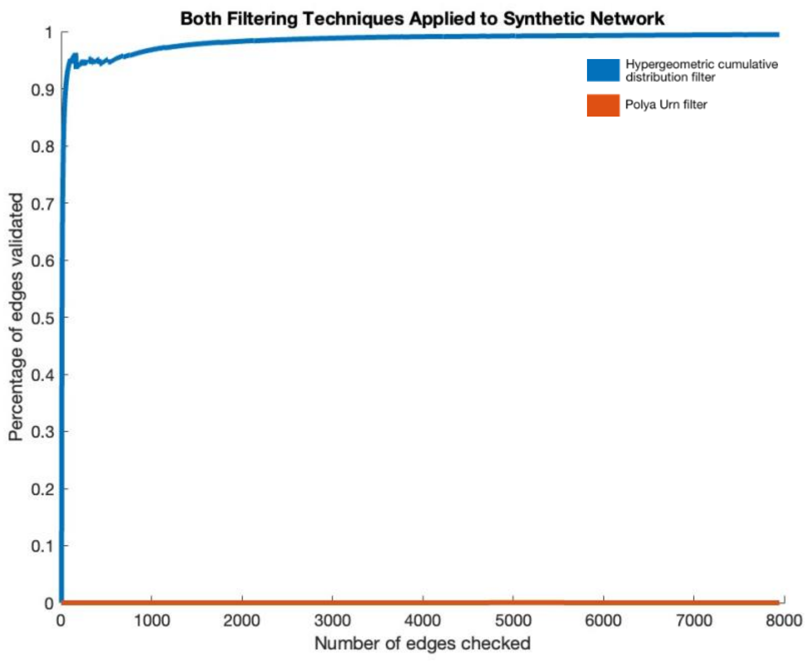

Figure 9: Both filtering techniques applied to synthetic network where "a=0.6" for the Polya Urn filter

As seen above, while the hypergeometric filter validates almost all edges, with 7901 out of 7942, the Polya Urn filter only validates 1 edge. This indicates that in with a controlled, uniform network, these methods work completely differently to one another. However, in order to be able to see more changes when alterations are made to a synthetic network, the “a” parameter of the Polya Urn filter can be tuned to allow for more edges to be retained in the backbone. This is due to one of the main advantages of the Barabasi Albert Model being synthetically replicating the growth and preferential attachment of real-life models, where the Polya Urn method uses the “a” parameter specifically to lessen the effect of this on the backbone validation. Therefore, by instead decreasing the “a” parameter to 0.1, in a network with 7942 edges, the hypergeometric filter will validate 7908 and the Polya Urn filter 958.

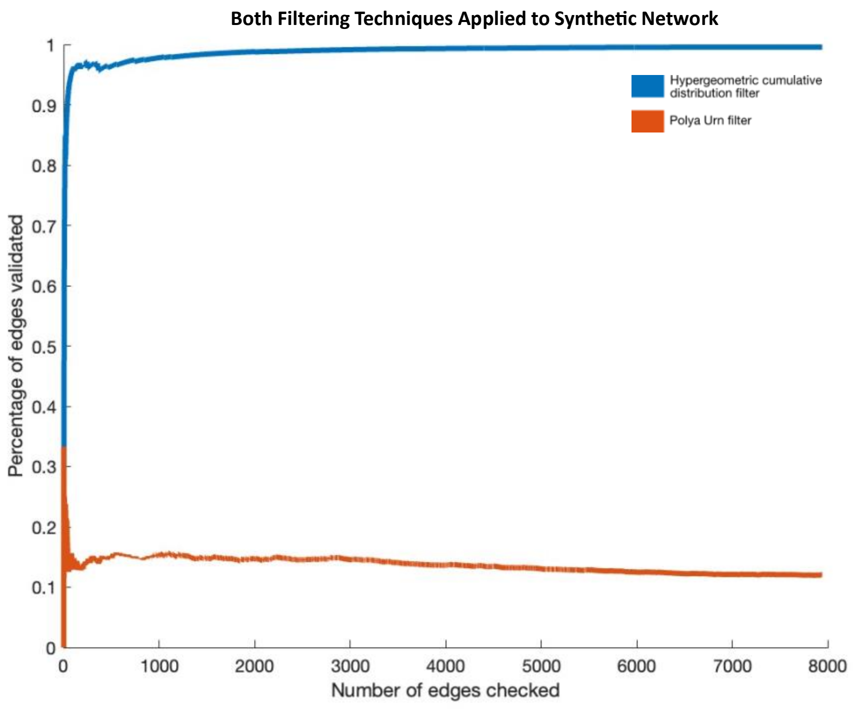

Figure 10: Both filtering techniques applied to synthetic network where "a=0.1" for the Polya Urn filter

4.3 Adding Fake Links to Existing Nodes

4.3.1 Comparing Altered Network to the Original Network

For the first alteration, fake links will be added to existing nodes in the network. In a network with 1006 nodes, nodes are selected at random, which in turn are given a random number of new edges in the network. This makes the uniform synthetic network more similar to real-life datasets, as small anomalies are present. These anomalies cause the network to produce differing statistical features, which are outlined below.

Table 5: Basic statistical features of a synthetically generated network before and after alterations

Network Statistic	Initial Network	After Alterations
Nodes	1006	1006
Edges	7932	9921
Average Degree	7.885	9.862
Network Diameter	5	2
Graph Density	0.008	0.01
Connected Components	1	1
Average Clustering Coefficient	0.035	0.32
Average Path Length	3.175	1.99

These changes clearly illustrate an increase in the average degree due to the increase in edges per node, which results in more connectedness throughout the network. It is important to note however, that only a small number of nodes have been given a large increase, which results in the growing connectedness of the whole network. This is further apparent by the network diameter, density, clustering coefficient and path length which all illustrate a network with a shorter flow of information between nodes.

4.3.2 Applying Filtering Techniques to the Altered Network

This easier flow of information through a network can further be analysed by seeing the effects this change has on the backbone extraction. When first looking at a single network which has been changed, there are distinct changes in the validation from the filtering techniques. This is seen from the percentage of nodes which are added to the backbone. For the hypergeometric filter, the validation has decreased from 99.6% to 90.5%. For the Polya Urn method the number of validated edges has increased from 11.7% to 13%. The percentage of edges validated by the filters is further shown below.

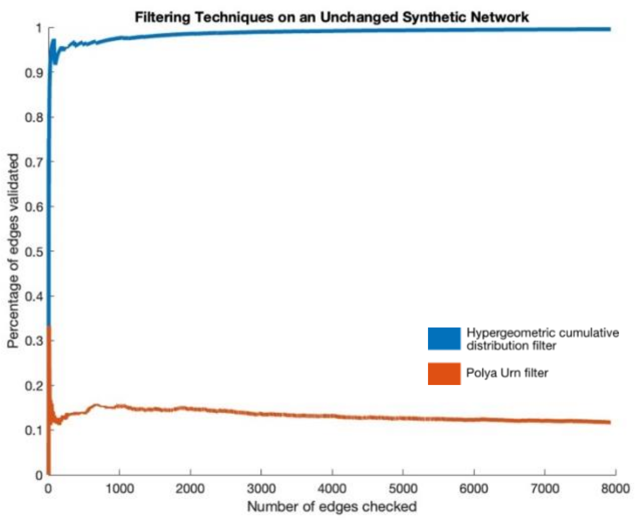

Figure 12: Both filtering techniques applied to the initial synthetic network

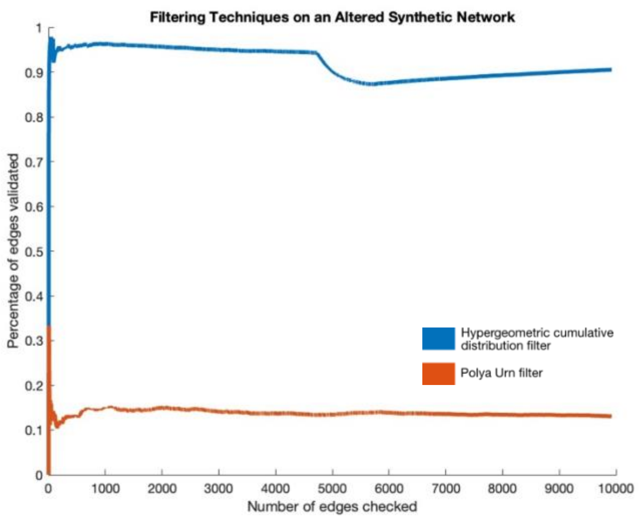

Figure 11: Both filtering techniques applied to the altered synthetic network

Lastly, it can be observed how the number of links validated by the filtering methods changes dependent on how many edges are given to random nodes. By gradually increasing the number of links added, the percentage of links validated to the backbone for each method can be calculated. This is demonstrated as follows.

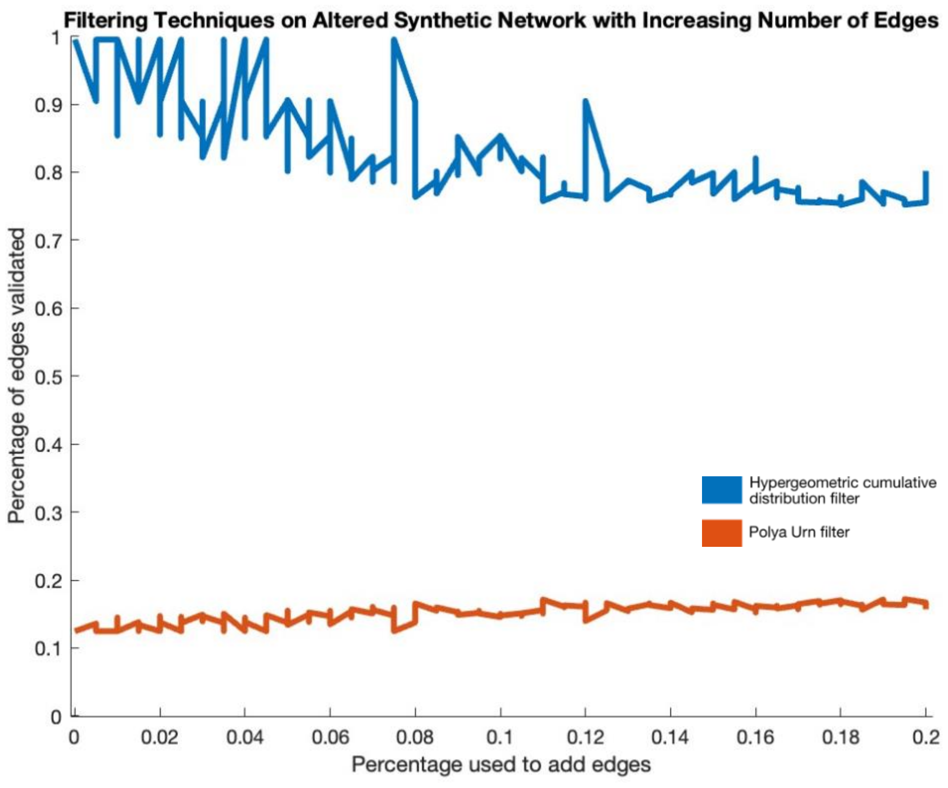

Figure 13: Percentage of validated links which changes with the number of edges added to existing nodes

As shown above, the percentage of validated links in the network changes drastically for both filtering techniques, dependent on how many new links are added to randomly selected nodes. This also coincides with what was seen when the two methods were compared using the airport network, where the hypergeometric filter removed nodes with larger degrees, where flights between major hubs were removed, while the Polya Urn filter retained most of those major hubs and their connections. Therefore, in a synthetic model where certain nodes have been converted into major hubs of the network, these same results are replicated.

Using a more homogenous power law distribution to draw edge weights, the two filtering methods react differently. Firstly, while there similarly is a trend where the percentage of edges validated decreases with the number of edges added for the hypergeometric filter, this is less noticeable when weights are drawn from a homogeneous distribution. The difference with the Polya Urn filter is however much larger, as no change is noticed regardless of the number of edges added to randomly selected nodes. Additionally, the Polya Urn filter validates almost no edges in this case at all, which also differs from the synthetic network with weights drawn from a heterogeneous distribution. The results from adding links to the network with homogeneously drawn weights are shown in the figure below.

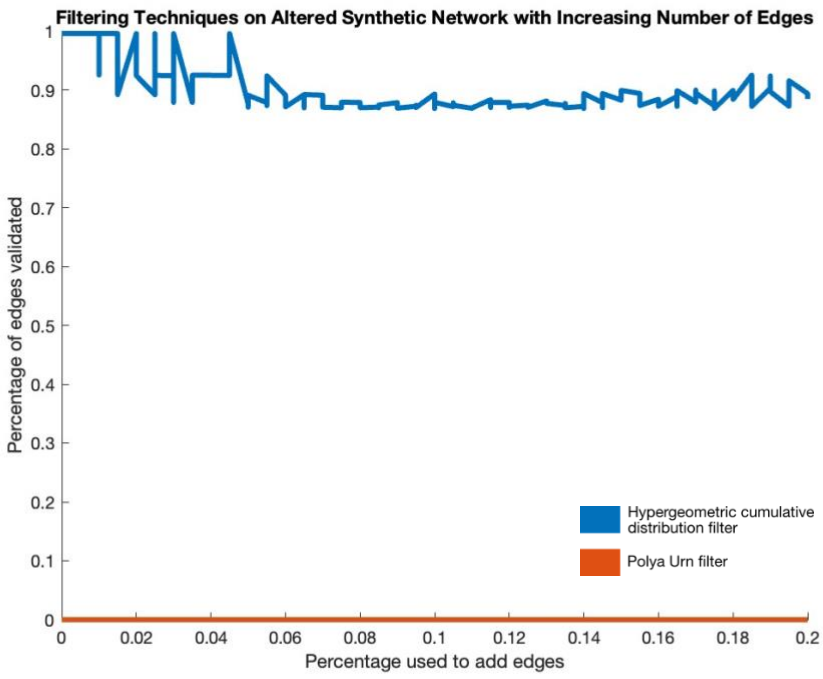

Figure 14: Percentage of validated links which changes with the number of edges added to existing nodes for a homogeneous weight distribution

4.4 Changing Weights of Links for Random Nodes

4.4.1 Comparing Altered Network to the Original Network

The second alteration which will be used to analyse how filtering techniques react to changes with synthetic data, is changing the weights of links for random nodes. Similar to the last change, this will create nodes which have a greater impact on the flow of information within the network. This is done by using a given probability to select nodes, which have their edge weights increased using a "gamma" multiple. As with the last alteration, this produces a network which should more effectively emulate a real-life system, which should be demonstrated by the backbone extraction methods. To first illustrate how this network differs from an unchanged synthetic dataset, a visual representation is shown below, where the edge weights are coloured from blue to red based on their weight. In the second diagram, there are clearly nodes with greater edge weights. The gamma multiple used in this example was set to 1.5 times the initial weight.

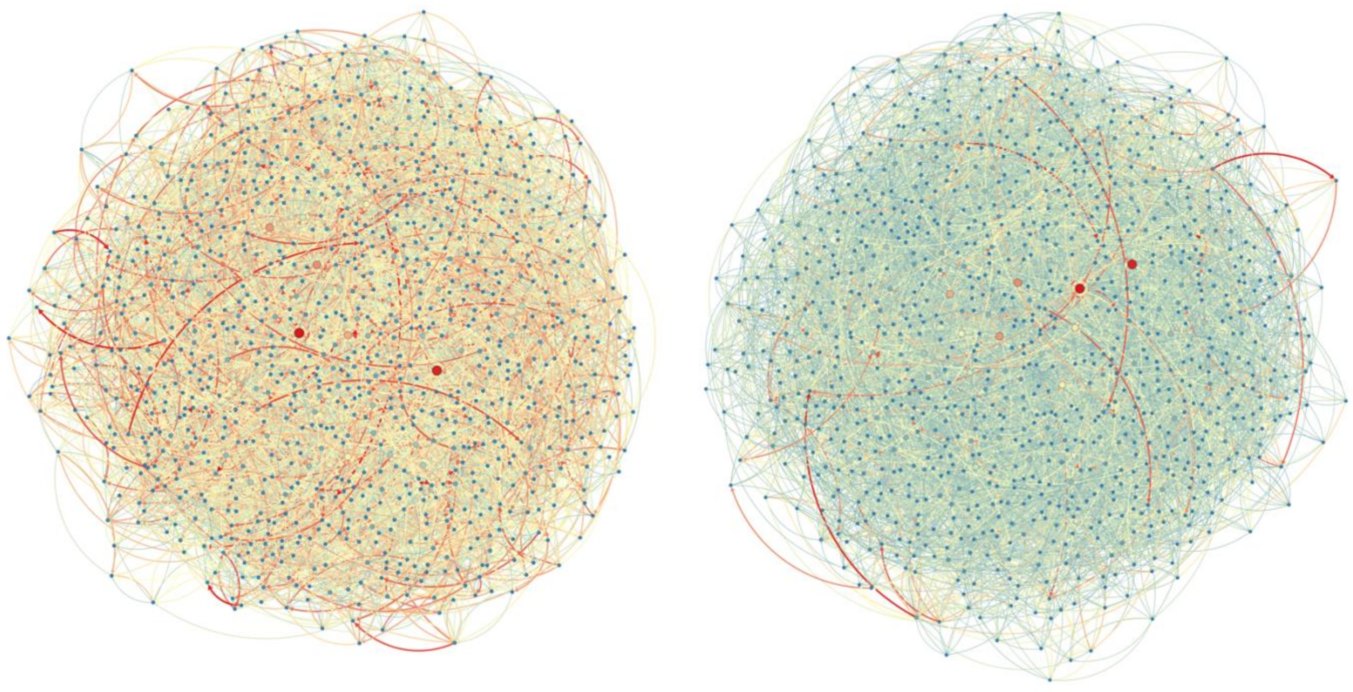

Figure 15: Visual comparison of synthetically generated network before and after increasing existing edge weights of random nodes

4.4.2 Applying Filtering Techniques to the Altered Network

For the network shown above, by multiplying the edge weights of random nodes by value of 1.5, the Polya Urn method and hypergeometric filter react differently. For the hypergeometric filter, barely any changes to the backbone have been detected, and the percentage of validated nodes remains almost identical, from 99.5% to 99.6%. The Polya Urn filter however does detect the anomaly in the data, and the percentage of validated links increases from 12.1% to 15.0%. The graph below further indicates this change in percentage of links validated for each method.

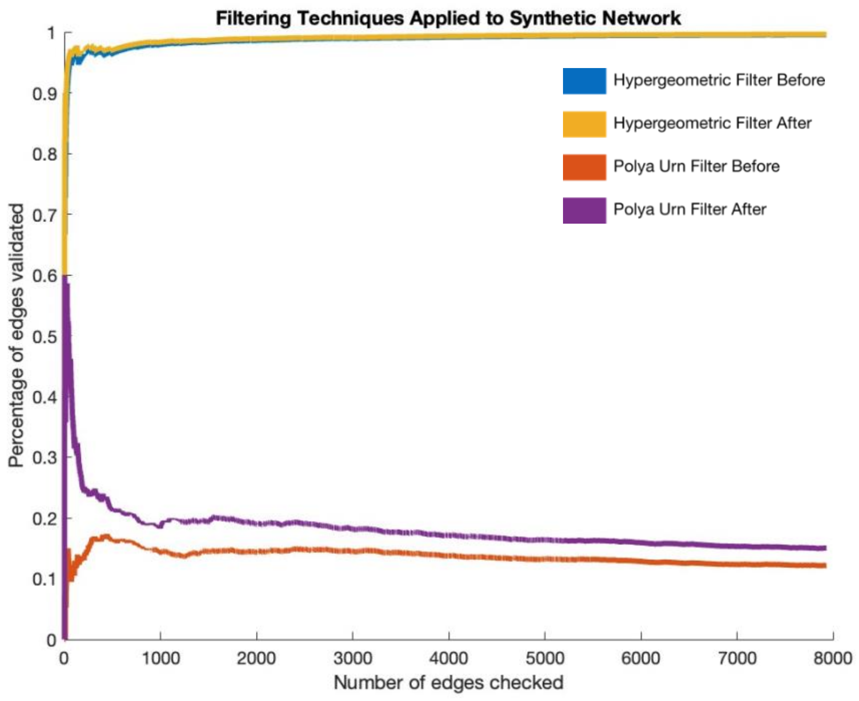

Figure 16: Filtering techniques applied to synthetically generated network before and after edge weights have been altered by a gamma multiple of 1.5, showing the percentage of edges validated over the number of edges checked

While the hypergeometric filter does not detect a large change having taken place when increasing the weight of randomly selected edges, it is interesting to note that a larger reduction in the edge weights is detected by both filters. By setting the gamma multiple to 0.02, resulting in altered edge weights to be near 0, there is a small reduction in the number of edges that the hypergeometric filter validates. In this case, the hypergeometric filter's ratio of validated links falls from 99.8% to 97.9%, and for the Polya Urn method this increases from 12.5% to 16.1%.

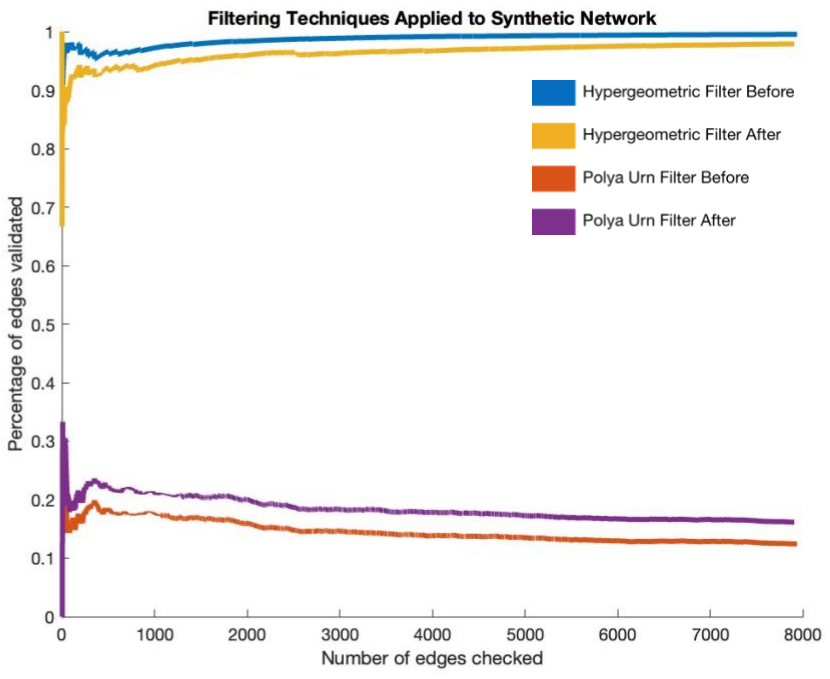

Figure 17: Filtering techniques applied to synthetically generated network before and after edge weights have been altered by a gamma multiple of 0.02, showing the percentage of edges validated over the number of edges checked

Lastly, by increasing the gamma multiple every time the filters run, the effect of the multiple can be accurately demonstrated. This is shown in the figure below, where 100 different gamma levels are tested between 0 and 5.

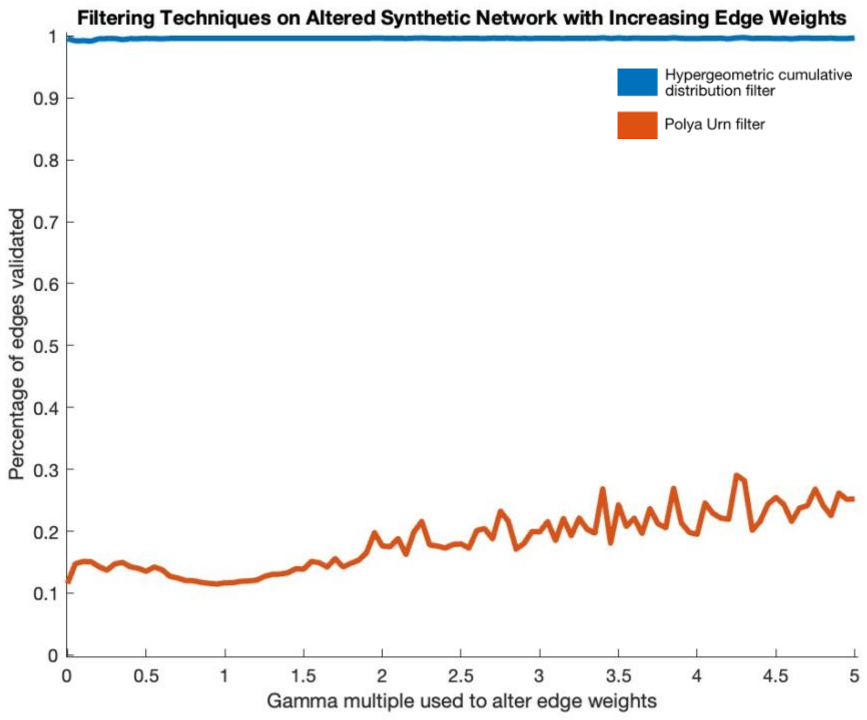

Figure 18: Percentage of validated links which changes with the edge weights given to existing nodes

It is interesting to note that similar to the first alteration that was made to the network, the percentage of links validated for both methods converge and become more similar to what was seen in the airport network. This can be attributed to the fact that introducing such an anomaly more accurately resembles real-life systems. While in this case the change made to the synthetic network had less of an effect on the validated links using the hypergeometric filter, the Polya Urn method showed similar results as when adding new edges for randomly selected links. Again, the fact that the altered network results look more like the results of the airport network could be due to the filters detecting more major hubs in the network, which the hypergeometric filter removes, and Polya Urn method retains from the backbone respectively.

When applying a different power law distribution where edge weights are drawn from a more homogenous curve, very similar results can be observed. This indicates that even with more or less variations in the initial edge weights, both filters can still detect even slight alterations to the weights of links, demonstrated by the extracted backbone. It is notable however that with edge weights from a more homogeneous distribution, the Polya Urn filter takes longer to react to a change. This could be due to the fact that if edge weights are already closer to one another, only a few outliers in the data do not have a considerable effect on the efficiency of information flow through the network, thus not validating more edges initially. However, using a higher multiple the filter does react to the alteration.

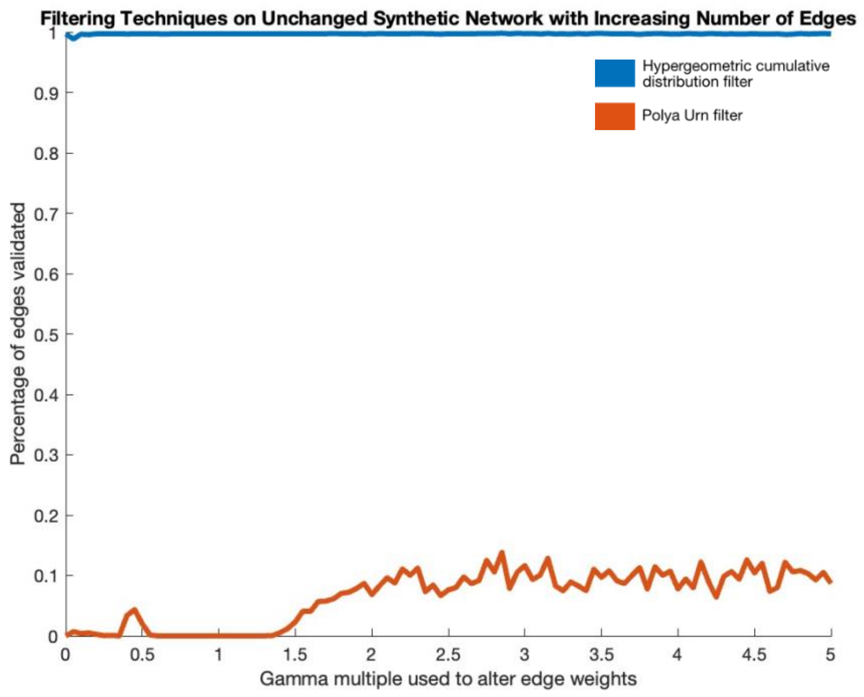

Figure 19: Percentage of validated links which changes with the edge weights given to existing nodes from homogeneous distribution

4.5 Reshuffling Link Weights at Random

4.5.1 Comparing Altered Network to the Original Network

The final alteration that is made to the synthetically generated networks is shuffling edge weights between nodes. This is done by taking the edge weights and assigning a random new position without adding or removing any links. This results in a network with the same basic network statistics as before shuffling, however with some edges gaining or losing importance in terms of edge weights.

4.5.2 Applying Filtering Techniques to the Altered Network

When applying both filtering techniques to first the unchanged network, and then the altered network, with a network built with heterogeneous edge weights slight differences are detected. For the hypergeometric filter the change in the percentage of links validated falls between -3.5% and 0.5%, while this has a greater effect on the Polya Urn filter, where the difference is between -6% and 8%, with both seeing the majority of percentage differences at

2%. These percentage were taken for the same changes of weights, over the course of 100 reshuffles. It could be concluded from this that the Polya Urn filter takes edge weights into greater account when calculating the backbone, where the hypergeometric filter uses more of the information related to the actual links between nodes. This confirms what was also seen with the other changes made to the synthetically generated networks. Two histograms of percentage differences over 100 trials are shown below.

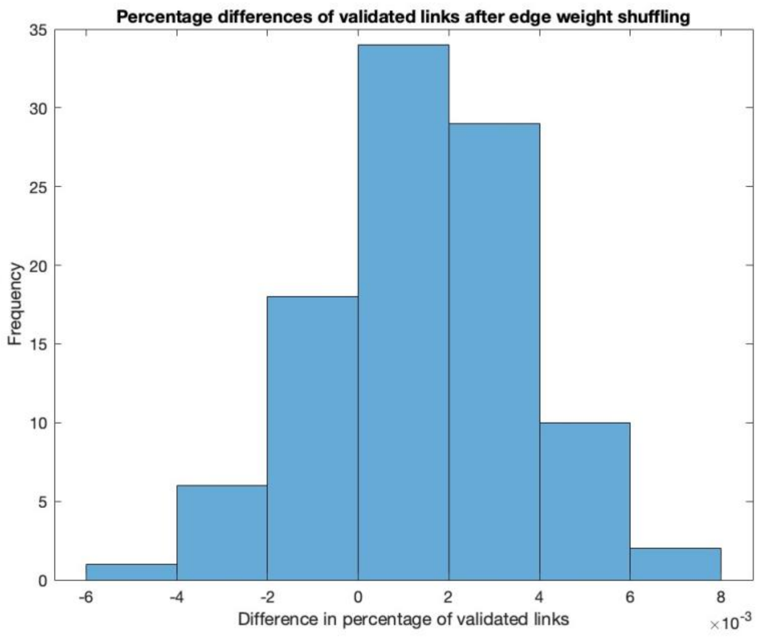

Figure 20: Histogram illustrating the difference in percentage of validated links for a changed synthetic network using the hypergeometric cumulative distribution filter

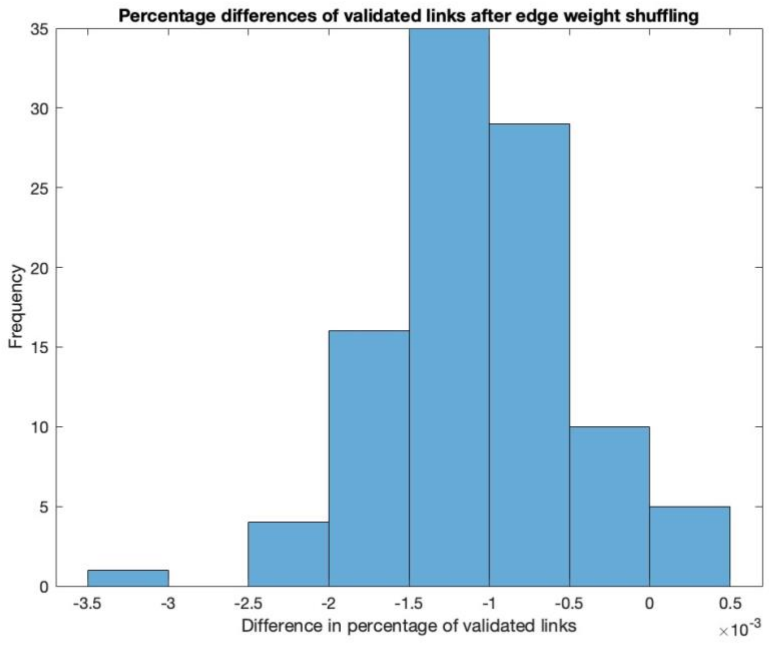

Figure 21: Histogram illustrating the difference in percentage of validated links for a changed synthetic network using the Polya Urn filter

When a more homogeneous power law distribution is used for drawing edge weights of the synthetic network, slight differences in the results are seen. Firstly, for the hypergeometric filter the results are similar with most trials having a 2% difference in the number of edges validated. What is different is that the spread is much larger, with percentage differences falling between -6% and 4%. With the Polya Urn filter, a similar result is seen as with the first alteration which was made to the synthetic network. When building a network with edge weights from a homogeneous distribution, almost no edges get validated. After this change had been made of reshuffling edge weights, the same is seen, as the main features of the network do not change. This results in no real changes between the unchanged and changed networks, as hardly any links are validated in both cases, meaning there is close to a 0% difference. Due to this, only the histogram for the percentage changes of the hypergeometric filter is shown below.

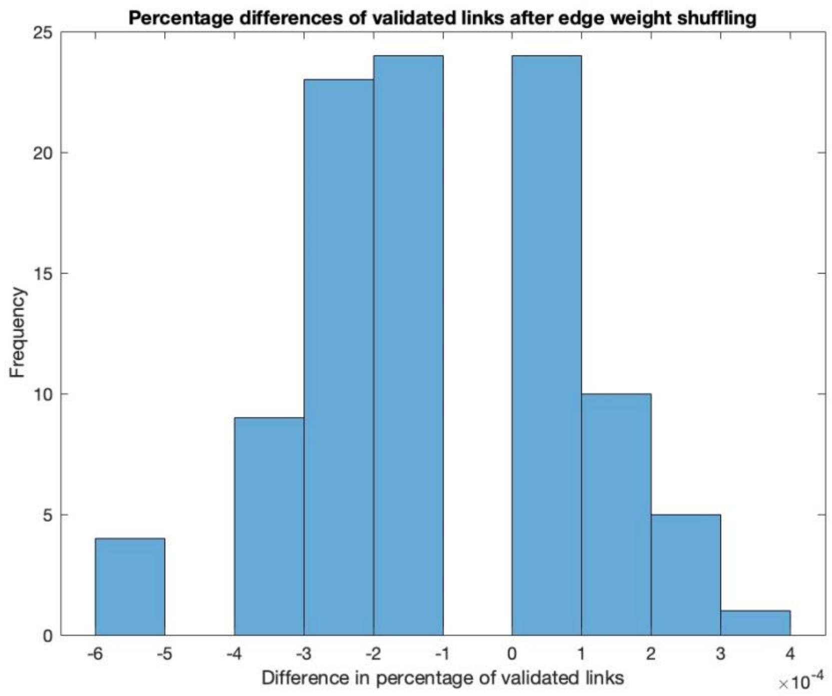

Figure 22: Histogram illustrating the difference in percentage of validated links for a changed synthetic network with edge weights drawn from a homogeneous distribution, using the hypergeometric cumulative distribution filter

Chapter 5: **Conclusion**

5.1 Critical Analysis and Achievements

The identification of relevant links in complex networks can be a challenging task but can provide information about the dataset which has many useful real-world applications. This task of extracting the backbone can be challenging however, and results vary based on the dataset used, and which filtering method was applied. This paper aimed to use a "ground truth" method to identify how filtering techniques differed, and what changes to a network make to the extracted backbone. The initial results when using the hypergeometric cumulative distribution filter and the Polya Urn filter on the Airport Network first provided an insight into how these two techniques differ. Through the generated outcomes, it could be concluded that firstly the methods had a different effect on major hubs in the network, where the Polya Urn Filter retained many of these connections, while the hypergeometric filter removed many flights between major hubs. This was further supported by the observation that the Polya Urn filter removed some nodes and all its flights completely, while the hypergeometric filter instead removed only certain flights, while keeping more nodes present. This was confirmed by the change in the network's graph density and the number of connected components.

These results obtained from the two filtering techniques about the Airport Network were then used to analyze the results of the filters on a synthetically generated network. What was first apparent when comparing the two methods on a uniform synthetic network was that they reacted oppositely. In the first trials, the hypergeometric filter validated nearly 99% of the links, while the Polya Urn filter validated close to 0%. However, by tuning the "a" parameter of the Polya Urn filter, it added over 10% of links to the backbone. These results could then be used as a benchmark for comparing the percentage of validated links after alterations had been made to the network.

Throughout the testing, three main network changes had been made. These were adding fake links to certain nodes, changing the weights of single links of nodes, and lastly reshuffling link weights at random. When looking at the results of the first two network alterations on a network where weights were drawn from a heterogenous distribution, it can be observed that the extracted backbones from the synthetic network more similarly resembled the results of the filters on the Airport network. This could be as there is rarely a case with real-world networks where they appear as uniform as the synthetic network. However, with the first two main changes made, some nodes were given more importance, and were turned into major hubs in the network. This resulted in the hypergeometric filter validating less links, while the Polya Urn filter validated more.

The information gathered from this is interesting when looked at in conjunction with the airport network backbones, as it can be concluded what changes to the synthetic network precisely made the results of the filters more resemble the real-life example. This allows for a better understanding of where each filter can be used more appropriately on real-life networks. For example, the increasing link weights had a much larger effect on the Polya Urn filter than for the hypergeometric filter. Thus, in a real-life example where two networks are compared with similar features but some nodes containing greater overall weights, the Polya Urn filter would be more effective than the hypergeometric filter in producing two different significant backbones for these networks. This shows that the use of a synthetic dataset to test changes on networks can provide an insight in the differences between filters which is otherwise hard to deduce.

For the last network alteration made, similar conclusions can be met. While both filters clearly detected an anomaly in the data when edge weights were shuffled, the changes in weights had a greater effect on the Polya Urn filter. From this, it can be concluded that in a real-life scenario, with a network where the importance more lies in the weights distributed among the links rather than the number of connections each node has, the Polya Urn filter would be the more ideal method to extract a backbone.

Lastly, these network changes also had differing effects on a network where weights were drawn from a homogeneous distribution. In many cases, the same results were observed for the hypergeometric filter, while the Polya Urn filter in almost all cases validated close to 0% of the links in the network. This was however not the case when link weights were altered for certain nodes. This indicates that when the Polya Urn filter is applied, it is important that there are greater differences between edge weights, otherwise the hypergeometric filter is a better option in identifying the relevant links.

As has been demonstrated by these tests done using the hypergeometric cumulative distribution filter and the Polya Urn filter on varying complex networks, many results can be gathered when comparing the outcomes. By initially testing the filters on the airport network, after which applying them to synthetic networks with various alterations made, the factors that affect the backbones calculated using these methods could be more accurately investigated. This provided an informative insight into which applications of the chosen filtering methods are more effective, and demonstrated how the use of synthetically generated datasets can improve the understanding of backbone extraction methods.

5.2 Further and Future Work

There are a couple of aspects which could still be investigated further in the future to provide more information on filtering techniques. First, more changes could be made not only to the synthetic network, but also to a real-life network to observe the differences in validated backbones. Furthermore, the changes made could cover more features of the networks, such as investigating what would happen when changing the network diameter, or when disconnecting edges and introducing more subcomponents in the system. Another change which could be made is comparing the synthetic network results to a greater number of other networks, apart from just the airport network. This could provide more information when altering the synthetic network, as it could be analyzed if results also resemble other features of real networks, apart from what was seen in the airport network.

Finally, what was learned about the filtering techniques could be applied to a real network, to observe whether the conclusions gotten from the synthetic network changes held up. For example, what was learned could be applied to the data of a logistics company, where nodes are warehouses, edges the transports between them, and weights the amount of cargo shipped. From the conclusions gathered about filtering techniques in this paper, and using some basic network statistics and features, an informed decision could be made on which technique would be most appropriate in that scenario. Such a case could then be used to demonstrate where a freight transportation company's most valuable shipping lines lie, in order to make better decisions on what area of the system to focus on. While these improvements are not essential in the current analysis done, in future they could provide further insights on the uses of backbone extraction methods.

References

1. Serrano, M.Á., Boguñá, M. and Vespignani, A. (2009) ‘Extracting the multiscale backbone of complex weighted networks’, *Proceedings of the National Academy of Sciences*, 106(16), pp. 6483–6488. doi:10.1073/pnas.0808904106.
2. Marcaccioli, R. and Livan, G. (2019) ‘A Pólya urn approach to information filtering in complex networks’, *Nature Communications*, 10(1). doi:10.1038/s41467-019-08667-3.
3. Serrano, M.Á., Boguñá, M. and Vespignani, A. (2009) ‘Extracting the multiscale backbone of complex weighted networks’, *Proceedings of the National Academy of*
4. Mastrandrea, R. *et al.* (2014) ‘Enhanced reconstruction of weighted networks from strengths and degrees’, *New Journal of Physics*, 16(4), p. 043022. doi:10.1088/1367-2630/16/4/043022.
5. Newman, M.E.J. (2019) ‘12’, in *Networks*. Oxford: Oxford University Press.
6. *Clustering coefficient in graph theory* (2022) *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/clustering-coefficient-graph-theory/ (Accessed: 22 August 2023).
7. *What is the error variance and how is it calculated?* (2023) *Study.com*. Available at: https://homework.study.com/explanation/what-is-the-error-variance-and-how-is-it-calculated.html (Accessed: 22 August 2023).
8. Coscia, M. and Neffke, F.M.H. (2017) ‘Network backboning with Noisy Data’, *2017 IEEE 33rd International Conference on Data Engineering (ICDE)* [Preprint]. doi:10.1109/icde.2017.100.
9. Dianati, N. (2016) ‘Unwinding the hairball graph: Pruning algorithms for weighted complex networks’, *Physical Review E*, 93(1). doi:10.1103/physreve.93.012304.
10. Brooks-Bartlett, J. (2018) *Probability concepts explained: Maximum likelihood estimation*, *Medium*. Available at: https://towardsdatascience.com/probability-concepts-explained-maximum-likelihood-estimation-c7b4342fdbb1 (Accessed: 22 August 2023).
11. Frost, J. (2023) *Marginal distribution: Definition & finding*, *Statistics By Jim*. Available at: https://statisticsbyjim.com/basics/marginal-distribution/ (Accessed: 22 August 2023).
12. *Hypergeometric distribution* (2023) *Encyclopædia Britannica*. Available at: https://www.britannica.com/topic/hypergeometric-distribution (Accessed: 22 August 2023).
13. Frost, J. (2023) *Hypergeometric Distribution: Uses, Calculator & Formula*, *Statistics By Jim*. Available at: https://statisticsbyjim.com/probability/hypergeometric-distribution/ (Accessed: 22 August 2023).
14. *HYGECDF* (2023) *Hypergeometric cumulative distribution function - MATLAB hygecdf - MathWorks United Kingdom*. Available at: https://uk.mathworks.com/help/stats/hygecdf.html (Accessed: 22 August 2023).
15. Armstrong, R.A. (2014) ‘When to use the Bonferroni correction’, *Ophthalmic and Physiological Optics*, 34(5), pp. 502–508. doi:10.1111/opo.12131.

16. Barabási, A.-L. and Pósfai, M. (2017) *Network science*. Cambridge, United Kingdom: Cambridge University Press.
17. *Preferential attachment - neo4j graph data science* (2023) *Neo4j Graph Data Platform*. Available at: https://neo4j.com/docs/graph-data-science/current/alpha-algorithms/preferential-attachment/#:~:text=History%20and%20explanation,work%20on%20scale%2Dfree%20networks (Accessed: 22 August 2023).
18. (2023) *Bureau of Transportation Statistics*. Available at: https://www.bts.dot.gov/ (Accessed: 22 August 2023).
19. *Ground truth* (1970) *Techopedia*. Available at: https://www.techopedia.com/definition/32514/ground-truth (Accessed: 22 August 2023).
20. *Ground truth* (2023) *MATLAB & Simulink*. Available at: https://uk.mathworks.com/discovery/ground-truth.html (Accessed: 22 August 2023).
21. Manager, O.Z., Zabolotnyy, O. and Manager, M. (2023) *A guide to training and testing data in machine learning*, *First Bridge*. Available at: https://firstbridge.io/blog/artificial-intelligence/training-and-testing-data-in-machine-learning (Accessed: 22 August 2023).
22. *Graph Adjacency* (2023) *Graph adjacency matrix - MATLAB adjacency - MathWorks United Kingdom*. Available at: https://uk.mathworks.com/help/matlab/ref/graph.adjacency.html (Accessed: 22 August 2023).
23. *Polya Filter* (2023) *MathWorks*. Available at: https://uk.mathworks.com/matlabcentral/fileexchange/69501-pf (Accessed: 22 August 2023).
24. Baeldung, (2022) *Computing the diameter of a network*, *Baeldung on Computer Science*. Available at: https://www.baeldung.com/cs/graphs-network-diameter (Accessed: 22 August 2023).
25. Geun Song a *et al.* (2017) *Analysis of the air transport network characteristics of Major Airports*, *The Asian Journal of Shipping and Logistics*. Available at: https://www.sciencedirect.com/science/article/pii/S209252121730041X (Accessed: 22 August 2023).
26. Datta, S. (2022) *Graph density*, *Baeldung on Computer Science*. Available at: https://www.baeldung.com/cs/graph-density#:~:text=Graph%20density%20represents%20the%20ratio,new%20edges%20to%20the%20network (Accessed: 22 August 2023).
27. Baeldung (2022) *Connected components in a graph*, *Baeldung on Computer Science*. Available at: https://www.baeldung.com/cs/graph-connected-components (Accessed: 22 August 2023).
28. Newman, M.E. (2006) ‘Modularity and community structure in networks’, *Proceedings of the National Academy of Sciences*, 103(23), pp. 8577–8582. doi:10.1073/pnas.0601602103.
29. *Clustering coefficient in graph theory* (2022) *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/clustering-coefficient-graph-theory/ (Accessed: 22 August 2023).

30. Bettilyon, T.E. (2019) *Types of graphs*, *Medium*. Available at: https://medium.com/tebs-lab/types-of-graphs-7f3891303ea8 (Accessed: 24 August 2023).
31. Embl-Ebi (2023) *Centrality analysis*, *Centrality analysis | Network analysis of protein interaction data*. Available at: https://www.ebi.ac.uk/training/online/courses/network-analysis-of-protein-interaction-data-an-introduction/building-and-analysing-ppins/topological-ppin-analysis/centrality-analysis/#:~:text=Centrality%20gives%20an%20estimation%20on,trying%20to%20find%20drug%20targets. (Accessed: 24 August 2023).
32. *Closeness centrality - neo4j graph data science* (2023) *Neo4j Graph Data Platform*. Available at: https://neo4j.com/docs/graph-data-science/current/algorithms/closeness-centrality/#:~:text=Closeness%20centrality%20is%20a%20way,distances%20to%20all%20other%20nodes. (Accessed: 24 August 2023).