Approaching Networks and Data

Sam Rosen

April 2019

Abstract

Big Data has become a coined phrase in the industry for large sums of information that have become available to Data Scientists. As more data becomes available on networks (infrastructure, social networks, etc.) algorithms to analyze them and draw conclusions have become in demand. This paper is a summary of multiple aspects to analyzing relational data that is best modeled with a graph. Each section contains a briefing of basic guidelines and considerations. The paper begins by introducing the idea of a graph and the presence of graphs in real data. The paper grows narrower in focus from abstract topics to specific measures of a network. The final sections include a briefing on multigraphs which have recently jumped in demand, an introduction to the systems one may use to process data with, and the mistakes and shortcomings to be expected in the process.

1 Introduction

A graph $G = (V, E)$ is a set of vertices $v \in V$ and a set of edges $e \in E$. Pairs of vertices can be connected by an edge $e = (u, v)$; $u, v \in V$. Undirected graphs give all edges $e = (u, v)$ a symmetric relationship without direction. Directed graphs contain edges that start from one node and end at another node, an asymmetric relationship. Graphs are the basis of a multitude of algorithms that are famously used to solve problems by companies such as Google and Facebook. For a larger introduction to graphs and theory surrounding them see [10].

Graphs have found a large presence in various communities for their ability to process data and draw conclusions. Graphs which are mapped to some sort of real data are called usually called networks. Network science is the study of networks. These networks can be tangible such as a road map or subway system or potentially more abstract describing relationships among friends on Facebook. Barabási, a large figure in the network science and physics community, notes network science is celebrated for its ability to explain high levels of complexity while maintaining broad theory for *most* needs [5]. Network science is an attempt to quantify the high levels of complexity composed by the effects of aggregated individuals in a network.

Examples of applications in the literature include modeling the interactions of proteins [2], analyzing the World Trade Network [6], and measuring discourse among political blogs [1] or a karate club [32]. Companies such as Google and Facebook maintain giant networks used millions of times a day by users. Google maintains a network where web pages are nodes and links from one page to another are directed edges (see Section 6). Facebook naturally has users as nodes and their friendships as links. Natural structures like the brain can be modeled as a network. A simple fruit fly brain network contains about 40 thousands vertices [28]; a human brain could easily approach up to a billion vertices if they were to correspond to neurons.

Consider any sort of system described above: multiple contain huge amounts of data and interactions that occur many times in a short time frame. Individual interactions in each system are often related e.g. a Facebook user makes a new post and one of their friends "likes" the post. This copious amount of data has been called *Big Data* by the industry. Demand for understanding networks is not only for building theory and understanding the properties of network that make it unique, but also creating algorithms that can scale to millions or billions of nodes. A highly active area of research, network science has strong potential to give a new understanding to data.

2 Data Semantics

Data points that are naturally related transform to a network for analysis. Cities are connected by highways and the internet is a large number of devices connected by bundles of wires. These networks have edge predicates defined in reality. Social networks have less tangible edges: a network of twitter users where edges point from user to user based on followers is intuitive but the edge relationship is not easily defined. What does it mean to follow someone on twitter? Does it mean a follower actively listens to the ideas of another user? Does it even mean a follower reads their tweets? Certain user nodes could be accounts that have not been accessed in several years; users may have forgotten they followed another user and never unfollowed them. To remedy this ambiguity an edge could occur between nodes if the corresponding user has "liked" or "retweeted" more than 5 of another users tweets. Network data is best taken in the context of the problem. Edges should be defined clearly and reasonable assumptions should be made to make conclusions from a network.

Data that does not have an inherent relationship can be made into a network if the edges are carefully defined. Consider a dataset of 500,000 Amazon products [22]: nodes could be individual products and edges can be drawn between them if they have been purchased in the same order. Edges could potentially be drawn if the products have a high overlap in their descriptions or titles. Edges could be drawn if a particular user has left a review on both the products. Interesting questions on this data could be:

- How can products be grouped on the home page to encourage sales?

- Which items should we advertise to users to turn the highest profit?
- What odd pairs of items in different sections (clothes, outdoor equipment, electronics, etc.) are typically ordered together?

The definition of the edge predicate affects the network structure, the eventual results, and how the results can be interpreted. If multiple types of relations are present on a set of data one could consider using a multigraph (see Section 7).

Weighting is a flexible way to make networks robust. If two nodes have multiple interactions one can give an edge a weight of some real value. Weights play an important role in both centrality and clustering and give another tool for solving problems.

An example of building a network for analysis from arbitrary data while maintaining reasonable context is clique tree learning used for analyzing high dimensional data [26]. A mutual information graph is built from categorical data where a node equates to an attribute and the graph is completely connected. The edges have weights corresponding to the normalized mutual information (NMI) of the attributes. The graph is then pruned where edges corresponding to low NMI (≈ 0) are removed and cliques are found in an effort to cluster the categorical data based on high attribute dependence. Although none of the data has a requirement to be relational, a network can be used to analyze it with reasonable definitions for edges and nodes.

3 Networks as Matrices

Any single-layer network can be represented using an adjacency matrix. Consider a network, $G = (V, E), V = \{1, ..., n\}$. An adjacency matrix A has entry $A_{i,j} = 1$ if and only if there is an edge from vertex i to vertex j, 0 otherwise. If a graph is undirected the matrix is symmetric. Matrices can be easily generalized for weighted graphs with $A_{i,j} = e_{i,j}$ where $e_{i,j}$ is the weight of the edge from vertex i to vertex j.

Adjacency matrices allow the domain of network analysis problems to extend past combinatorics and discrete structures into the domain of matrix analysis. Matrix analysis contains an array of tools that easily apply to network problems. Doubly-stochastic matrices are the convex hull of permutation matrices and are used to solve graph matching problems [28]. A framework for estimating matrices via the singular values has been applied to fitting stochastic block models and graphons (see Section 4) [11]. Treating adjacency matrices and their offshoots as linear maps is a successful technique for creating bounds on network behaviour. As a linear map, ideas such as product spaces and norms are well understood [21].

One of the strongest tools used is the laplacian matrix. It is defined as $L = D - A$ where D is a diagonal matrix of the vertex degrees and A is the adjacency matrix. The normalized laplacian is defined similarly with $L = I - D^{-1/2}AD^{-1/2}$. The spectrum of a laplacian matrix gives information on the

structure of a graph, including its clustering. The Courant-Fischer theorem gives a definition for Rayleigh quotients on the laplacian, allowing for answers to partitioning a network in reasonable fashions [12] [18].

A particularly convincing argument for the intrinsic relationship between networks and matrices is the association between eigenvalues of the laplacian and the connectivity of the graph. A graph is connected if and only if the second smallest eigenvalue of the laplacian of the graph is greater than 0 [31]. The ability to give a hard fact about a network by analyzing the spectrum will motivate many practical algorithms on networks (see Section 5,6).

4 Random Graph Models

A random variable $b \sim Bernoulli(p)$ follows a Bernoulli distribution with parameter $p \in [0, 1]$. Bernoulli random variables are simple: an event happens with probability p and does not happen with probability $1 - p$. Random graph models build upon Bernoulli random variables where the event of an edge between some two nodes is a typically an independent Bernoulli random variable.

The Erdős–Rényi model is the simplest of the random graph models. The model has two parameters: n, the number of nodes and p the probability of an edge between any two nodes. Each edge is independent of all other edges. The expected number of edges is $p \cdot \binom{n}{2}$. Although simple they contain a variety of interesting properties.

The Stochastic Block Model (SBM) can be thought of as a composition of k interacting Erdős–Rényi models. The SBM takes a number of communities k to model. The second parameter is a $k \times k$ symmetric matrix (for undirected models), P where $P_{i,j}$ is the probability there is an edge from a vertex in community i to a vertex in community j. $P_{i,i}$ is the probability an edge occurs between two vertices in community i. If P is a matrix with all equal entries then the SBM simplifies to a single Erdős–Rényi model. Each edge in an SBM is independent from the others. Stochastic block models are more robust than Erdős–Rényi models and are good for modeling community structure that occur often in social networks [17].

Generalizing further, we see an example of a latent position graph: the random dot product graph (RDPG). The most general definition includes some probability distribution for random unit vectors of dimension d. Each vertex becomes associated with a vector and the probability of an edge between two vertices is the magnitude of the dot product of their associated vectors. This configuration allows for large generality by defining the distribution over the vectors. The vectors can also be fixed to simplify to other models such as a stochastic block model [4].

By fitting a network to a random graph model one can describe in simple terms the interactions of communities in the data. As clear cut models with backing theory random graph models have a wide range of applications from community detection in social networks to detecting specific structures in brains [4]. Although we've covered three models here, there are many more in the

literature such as the Barabási-Albert model [3], the Bianconi-Barabási model [7], and the Watts–Strogatz model [30].

Notice all the specified models above assume all random variables are independent and identically distributed. Assuming independence is often a requirement for building theory and giving bounds and guarantees for a model. In practice this often works well but in certain contexts assuming independence may be unreasonable. If there is an edge from node a to node b and an edge from node b to node c some contexts would imply a much higher chance for an edge from node a to node c. A model which found success with the above considerations is the mixed stochastic block model [2]. The authors found success measuring contexts by using a offshoot of an expectation-maximization algorithm to estimate the posterior as sampling was intractable.

5 Clustering

A cluster or community is a set of vertices that are tightly coupled by some indicative structure. Given a network representing any sort of data, the ability to cluster certain nodes together has many applications. By splitting a large network into well-defined clusters the system could be understood as a much smaller set of clusters that interact with each other.

Clustering is a useful abstraction with no hard definition. Some hard definitions include modularity, a measure of link density inside clusters compared to between them [8]. The Louvain method is one of the first and most effective algorithms for clustering in large networks aimed to optimize modularity [8]. As a greedy algorithm it was able to derive useful communities on graphs with over 100 million nodes and a billion edges.

Conductance is a similar definition measuring density within communities as compared to without. The choice between the two comes with an effort for proving guarantees of the desired algorithm. In [8] the modularity definition allowed for quick easy computations of change in modularity while in [14] the definition of conductance allowed for a framework to give guarantees on approximate personalized page rank. The viability of clusterings may be compared by conductance or modularity but as an unsupervised method there is no definitive metric to determine if a clustering is the ground-truth.

Algorithms that have optimized the above measures give good approximations but can occasionally fall short. By computing some predetermined number of eigenvalues and eigenvectors from some derivation of the adjacency matrix one can cluster the graph nodes by a classic vector clustering method such as Gaussian Mixture Models or K-Means. This is known as spectral graph clustering (SGC). SGC plays upon the idea of cutting a graph into clusters or potentially showing areas that a random walk may struggle to leave from [29]. Spectral methods are typically more expensive but can often throw away noise in a graph for signal [12].

A good clustering can often answer questions on a network related to:

- How does community A overlap with community B?

- What's the smallest number of communities we can group a network into reasonably?
- Is there a non-obvious cluster in the graph of interest?

6 Centrality

Centrality is a general measure for the stature or "importance" of nodes in a network. A high centrality implies a high "importance" as the node is considered more central than others in the graph. Centrality is a flexible term and its implications depend on the chosen definition.

The most basic type of centrality is degree centrality. The degree centrality of a node is simply the degree of the node or the number of incident edges. Directed graphs can generalize this to in-degree and out-degree. This is a relative measure of centrality. If a node has a degree centrality of 100 this could be an anomaly if most nodes have a degree of ≈ 10 or ≈ 1000 but if this is a standard degree for the given graph it may not be an interesting measure. Degree centrality could be normalized but this could be skewed by nodes of extreme degree. Degree centrality can be generalized for weighted graphs where the measure is a sum of the weights of the edges; this is a decision to be made based on the context of the data. This simple definition has utility but it does not consider many aspects of the structure of a graph. However, the distribution of degree centrality gives a good overview of the overall graph structure. The degree distribution is often used to determine if a matrix is sparse [21] [11] giving guarantees for certain algorithms to succeed. If a network follows a power-law degree distribution one can derive insight to the scaling of the network or viability to be fit by a certain model [3].

If paths, particularly optimal ones, have strong implications on a network (typically infrastructure), betweenness centrality is valuable. The betweenness of a node is defined as the ratio of shortest paths containing v for which v is not a start or destination to the total number of shortest paths for which v is not a start or destination. Nodes with a high betweenness centrality are likely bottlenecks in a graph. Calculating betweenness is not tractable for large graphs due to a runtime of $\Omega(|V|^3)$. This overview was largely derived from Brandes [9] and his derivation for a faster algorithm to calculate betweenness on sparse graphs.

The spectrum of the adjacency matrix can also be used to give reliable measures of centrality. Spectral methods maintain utility via their justification and guarantees through matrix analysis. Eigenvector centrality derives from a simple idea: a node that is central should be connected to other central nodes [16]. Consider the equation: $c(v) = \frac{1}{\lambda}\sum_{i \in neighbors(v)} c(i)$ or $c(v) = \frac{1}{\lambda}\sum_{i \in neighbors(v)} e_{i,v}c(i)$ where $e_{i,v}$ is the weight of the edge from node i to node v. If one considers the function c as a vector over the nodes the equation can reduce to an eigenvalue problem: $A\vec{c} = \lambda\vec{c}$, where A is the (weighted) adjacency matrix and $\vec{c}$ is a vector with $\vec{c}_i$ as the centrality of node i. The

eigenvector corresponding to the largest eigenvalue is used as it is guaranteed to have all positive entries (via Perron-Frobenius theorem) and correspond to the best rank-1 approximation of A [16]. As an eigenvalue problem one can use any sort of iterative technique to produce the eigenvector for the highest eigenvalue.

A famous example of considering data context to perform analysis occurs in the original PageRank paper [24]. PageRank is an off-shoot of eigenvector centrality famously implemented by the founders of Google. The authors were able to derive high quality results by considering centrality in the highly specific context of web pages. Pages with a high PageRank are deemed important to the web and are more likely to occur at the top of search results.

7 Multilayer Networks

Consider two networks side-by-side each modeling a different system. If these systems were related one might want to model the networks together. This becomes a challenge as the edges in the two networks could have different meanings. The nodes in each network may or may not correspond to an equivalent object in the other network. Multilayer networks are an attempt to derive information from related networks. A multilayer network can generalize to any number of networks modeled side-by-side and can even generalize to multiple dimensions of coupled networks called aspects [19].

Graphs have the ability to model complex systems but can fall short when modeling multiple coupled systems. A multilayer network can allow for more reasonable analysis of systems that are deeply integrated such as those in percolation theory. A simple example would be a two-layer graph where one layer models the spread of a virus and the other layer models the spread of information about the virus [19]. Multilayer networks may also be a way to reason about previously studied problems such as modeling a corpus with keywords [19].

The literature surrounding multigraphs is limited, but expanding rapidly. [19] is a survey of existing literature and an argument that multigraphs have large potential for future application. Generalizations of the random walk, Erdős–Rényi model, and centrality are also mentioned in [19] although much more is to be understood about these generalizations. Multigraphs can be modeled analytically with tensors. Although this representation is useful, tensors are harder to use than matrices. As an example, computing the rank of a matrix is simple to do efficiently, but computing the rank of a tensor is an NP-HARD problem. [20].

Given the lack of literature and difficultly of multilayer networks a suitable alternative may be useful. If a multilayer network can be easily mapped to a single network with different edge colors, weights can be assigned to each edge based on the color in the context of the problem. For example, in a social network problem the edges corresponding to friends could have weight 2 and the edges corresponding to family could have weight 10. The adjacency tensor of a multilayer network can be unwound into a matrix called a "supra-adjacency

matrix" [19]. This can be generalized to a "supra-laplacian" and matrix analysis can be used to analyze the multilayer network. Some groups have had success averaging over the adjacency matrices of the layers [12]. Success has been found in understanding the multinetwork of international trade by analyzing each layer individually and comparing to an aggregated graph [6]. Although aggregation methods have proved useful, signal maintained in the multilayer network is lost [19]. Careful consideration should be used when deciding to use a multilayer network as they can potentially give important insight missed in a typical network.

8 Computation

Although adjacency matrices allow for strong analysis of networks they are not the most efficient way to store graphs. Adjacency matrices require $O(|V|^2)$ space giving poor scaling for large graphs. Sparse adjacency matrices can be stored and processed faster with existing sparse matrix technology but this can be impractical when dealing with networks. Some networks may not warrant use of a sparse matrix if the graph is mostly filled with various edge weights. Sometimes when dealing with data it is possible to predict the graph structure well [30], but often the structure of the graph is unknown until the graph is stored on a system and analyzed. Small graphs can warrant use of an adjacency matrix with little problem, but some systems which scale to large graphs can handle small graphs just as well.

Large amounts of data are often stored on a distributed system as it is not feasible to store everything on a single hard drive or SSD. A distributed system distributes the data across multiple machines, often called a cluster. Some database manager controls queries executed to the system which collects results from each machine. A good partition is essential for the efficiency of a distributed system so any underlying management can quickly find data by exploiting locality and not look through each machine [13]. Enormous graphs, like enormous data, are often stored on a distributed system. The graph is stored as a composition of a subgraph on each machine with bridges (edges) between them. A good partition will minimize the number of bridges between the subgraphs allowing for efficient traversals and queries on the graph. If a call to the graph data requires traversal of a bridge additional processing needs to be done to access the adjacent machine and consolidate the information.

Distributed systems are natural environments for parallel algorithms. Parallel algorithms have a focus on efficiency in both time and resources giving rise to practically solving problems on distributed graphs. The ability to make an algorithm parallel can make an algorithm tractable on exceptionally large graphs ($|V| \gg 10^6$) [27] or exceptionally hard problems possible on moderately sized graphs ($|V| \approx 10^5$) [23].

Traditional databases have been relational. Relational databases are typically tables with items as rows and attributes as columns. There are considered relational as some columns contain id numbers that correspond to other data

items, potentially in another table. This design paradigm often lacks compatibility with graphs:

- Graphs and their algorithms typically require flexible data. It is not uncommon to have a network where there are several types (colors) of nodes each with different attribute sets and several types (colors) of edges also with different attribute sets. A relational database requires careful planning beforehand but graphs may be too complicated to spend the time mapping out an entire database.
- Relational databases are effective in many applications but typically results are returned after a single query. Graphs often require multiple queries to do basic algorithms such as path finding or counting the degrees of all neighbors. A relational database would require a number of queries proportional to the size of a path or neighbor set while a graph database can likely give these results in one optimized query.

Recently there have been developments to produce databases based on graphs that follow the NoSQL paradigm. NoSQL databases are not relational nor organized in tables. NoSQL databases are typically flexible to data writes and are more adept to distribution. By using a NoSQL database based on a graph, one can easily use a system that is adapted to graph structures for their graph algorithms. Some popular graph databases include: Neo4j [1], Blazegraph [2], and HBase [3].

In an effort to consolidate the various graph databases the Apache Software Foundation developed Gremlin [4]. Gremlin is an implementation independent traversal language for graphs. Although a steep learning curve is present, Gremlin has the potential to be an efficient solution to network analytics. Any graph database can be supported in Gremlin if a proper interface between the two are built. This allows an analyst to use a system that is suited towards their needs. Some specific justifications include:

- Neo4j: A fast backend for large graphs with support for various useful graphical user interfaces.
- Blazegraph: A backend based on the GPU potentially giving huge speedups, particularly for parallel algorithms.
- HBase: A backend with strong support for distributed data of enormous proportions.

9 Pitfalls of Network Science

Although network analysis can derive helpful conclusions, some mistakes can cause false conclusions to be drawn.

[1] `https://neo4j.com/`
[2] `https://www.blazegraph.com/`
[3] `https://hbase.apache.org/`
[4] `https://tinkerpop.apache.org/gremlin.html`

The definition of sparsity varies throughout the literature: sometimes it is defined as a bounded expected degree [21] and other times as an expected degree that grows logarithmically with the number of vertices [15]. It's somewhat disingenuous to make a choice of algorithm for analysis based on definitions of sparsity with grey areas. Analyzing the degree distribution is useful to give an idea of density, but it is ambiguous to claim any graph is sparse as the behaviour of the expected degree could potentially fall on either side of sparse or dense. The expected degree of a network asymptotically can only be estimated and could easily be close enough to the bound that it is hard to say which side it falls on. This does not bring confidence in conclusions drawn from algorithms that have a sparsity vs. dense requirement if a network is ambiguously dense.

Clustering is not a definitive term and although measures such as conductance and modularity can be used to compare them, a low measure does not indicate a bad clustering. Many network clustering papers have used datasets such as political blogs [1] with a given truth and claim a method is not viable if the clusters do not match the truth very well. Recently a paper has shown a comparison of two closely related clustering algorithms giving two different, but correct clusterings [25]. Existing clustering algorithm papers typically do not explain the structure of clusters extracted making it unclear which algorithm to use to extract a certain feature of a network.

Occasionally algorithms run on networks will yield good results but have very little backing theory to them which can be used to give guarantees on runtimes or accuracy [8] [14] [24]. Without rigorous theory behind the algorithm, results achieved could be wrong as certain criteria for convergence or precision were not met. Generally if an algorithm has been applied many times over with success it should be reasonable to use for analysis; however, recent algorithm developments without a convincing argument for correctness should be open to question.

Network analysis has yielded great results, but it is not an end-all be-all technique. It is perfectly reasonable to have data which has no place being put in a network (although a probabilistic graph model may be useful). If data points have no relationship between each other and no sensible edge relation can be derived, a network will yield no additional information and is likely drawn by some arbitrary unimportant metric. Graphs are definitely an exciting research area [5] but there will always be utility for other advanced methods for clustering, anomaly detection, and regression.

To avoid creating arbitrarily drawn networks it may be useful to consider what the following typical network structures imply about the data:

- Path: If a network has a path from a node s to another node t, what can be said about the relationship between the two entities corresponding to the nodes? How does this relationship get stronger or weaker as the path length increases?

- Clique: If some number of nodes are (almost) completely connected, what

[5] `https://www.wired.com/insights/2014/03/graph-theory-key-understanding-big-data/`

does this tight cluster imply about each of the nodes?

- Connected Component: If a graph is disconnected where some set of nodes have no path to another set of nodes, what is the distinction between the sets of nodes?
- Bottleneck: If two communities are present in a graph but have little inter-community edges, can something be said about the communities or the edges and nodes on the bottleneck?

10 Conclusion

Network analysis has become a highly cross-discipline area taking ideas from computer science, applied math, and statistics. Approaching problems that could draw meaningful conclusion through network analysis is difficult since the complexity of the model becomes difficult to navigate. Understanding the construction of the network allows a user to form questions that have useful answers. The various tools for understanding networks can be summarized as questions:

- Random graph models: How can my network be modeled accurately by a well-understood distribution?
- Centrality: Which nodes are particularly more important than others?
- Clustering: What and where are the communities in my graph?

The context of a problem may warrant a multilayer network due to the potential to carry more information. Lastly, to feasibly consider running algorithms on large networks, a system for computation needs to be prepared.

Although this guide was meant to give a strong introduction to network analysis the literature is vast. Some topics not covered in this write-up that hold large utility in network models are path-finding algorithms, combinatorial methods, and random walks. Some essential parts of an analytic pipeline that were not covered include integrating an additional data set with a network and using existing machine learning methods together with a network to draw conclusions.

References

[1] L. A. Adamic and N. Glance. The political blogosphere and the 2004 us election. *WWW-2005 Workshop on the Weblogging Ecosystem*, 2005.

[2] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *arXiv e-prints*, page arXiv:0705.4485, May 2007.

[3] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74:47–97, January 2002.

[4] Avanti Athreya, Donniell E. Fishkind, Keith Levin, Vince Lyzinski, Youngser Park, Yichen Qin, Daniel L. Sussman, Minh Tang, Joshua T. Vogelstein, and Carey E. Priebe. Statistical inference on random dot product graphs: a survey. *arXiv e-prints*, page arXiv:1709.05454, Sep 2017.

[5] Albert-László Barabási. The network takeover. *Nature Physics*, 8:14–16, 2011.

[6] Matteo Barigozzi, Giorgio Fagiolo, and Diego Garlaschelli. Multinetwork of international trade: A commodity-specific analysis. *Physical Review*, 81(4):046104, Apr 2010.

[7] G. Bianconi and A.-L. Barabási. Competition and multiscaling in evolving networks. *EPL (Europhysics Letters)*, 54:436–442, May 2001.

[8] Vincent D Blondel, Jean loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks, 2008.

[9] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25:163–177, 2001.

[10] Gary Chartrand. *Introductory Graph Theory*. Wiley-Interscience, 1984.

[11] Sourav Chatterjee. Matrix estimation by Universal Singular Value Thresholding. *arXiv e-prints*, page arXiv:1212.1247, Dec 2012.

[12] Pin-Yu Chen and Alfred O. Hero. Multilayer Spectral Graph Clustering via Convex Layer Aggregation: Theory and Algorithms. *arXiv e-prints*, page arXiv:1708.02620, Aug 2017.

[13] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. In *OSDI'04: Sixth Symposium on Operating System Design and Implementation*, pages 137–150, San Francisco, CA, 2004.

[14] Kimon Fountoulakis, Farbod Roosta-Khorasan, Julian Shun, Xiang Cheng, and Michael W. Mahoney. Variational Perspective on Local Graph Clustering. *arXiv e-prints*, page arXiv:1602.01886, Feb 2016.

[15] Sainyam Galhotra, Arya Mazumdar, Soumyabrata Pal, and Barna Saha. The Geometric Block Model. *arXiv e-prints*, page arXiv:1709.05510, Sep 2017.

[16] Peter Hoff. Centrality – 567 statistical analysis of social networks.

[17] Paul Holland, Kathryn Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks - SOC NETWORKS*, 5:109–137, 06 1983.

[18] Andre Wibisono Jonathan Kelner. 18.409 an algorithmist's toolkit; lecture 3. https://ocw.mit.edu/courses/mathematics/18-409-topics-in-theoretical-computer-science-an-algorithmists-toolkit-fall-2009/lecture-notes/MIT18_409F09_scribe3.pdf.

[19] Mikko Kivelä, Alexandre Arenas, Marc Barthelemy, James P. Gleeson, Yamir Moreno, and Mason A. Porter. Multilayer Networks. *arXiv e-prints*, page arXiv:1309.7233, Sep 2013.

[20] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM REVIEW*, 51(3):455–500, 2009.

[21] Can M. Le, Elizaveta Levina, and Roman Vershynin. Sparse random graphs: regularization and concentration of the Laplacian. *arXiv e-prints*, page arXiv:1502.03049, Feb 2015.

[22] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[23] Vince Lyzinski, Daniel L. Sussman, Donniell E. Fishkind, Henry Pao, Li Chen, Joshua T. Vogelstein, Youngser Park, and Carey E. Priebe. Spectral Clustering for Divide-and-Conquer Graph Matching. *arXiv e-prints*, page arXiv:1310.1297, Oct 2013.

[24] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web, 1999.

[25] Carey E. Priebe, Youngser Park, Joshua T. Vogelstein, John M. Conroy, Vince Lyzinski, Minh Tang, Avanti Athreya, Joshua Cape, and Eric Bridgeford. On a 'Two Truths' Phenomenon in Spectral Graph Clustering. *arXiv e-prints*, page arXiv:1808.07801, Aug 2018.

[26] Cetin Savkli, J. Ryan Carr, Philip Graff, and Lauren Kennell. Bayesian Learning of Clique Tree Structure. *arXiv e-prints*, page arXiv:1708.07025, Aug 2017.

[27] Soumendu Sundar Mukherjee, Purnamrita Sarkar, and Peter J. Bickel. Two provably consistent divide and conquer clustering algorithms for large networks. *arXiv e-prints*, page arXiv:1708.05573, Aug 2017.

[28] Daniel L. Sussman, Vince Lyzinski, Youngser Park, and Carey E. Priebe. Matched Filters for Noisy Induced Subgraph Detection. *arXiv e-prints*, page arXiv:1803.02423, Mar 2018.

[29] Ulrike von Luxburg. A Tutorial on Spectral Clustering. *arXiv e-prints*, page arXiv:0711.0189, Nov 2007.

[30] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. , 393:440–442, June 1998.

[31] Eric W Weisstein. Algebraic connectivity. From MathWorld–A Wolfram Web Resource. `http://mathworld.wolfram.com/AlgebraicConnectivity.html`.

[32] W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.