12-16-2022

Interpretable Network Representations

Shengmin Jin

Abstract

Networks (or interchangeably graphs) have been ubiquitous across the globe and within science and engineering: social networks, collaboration networks, protein-protein interaction networks, infrastructure networks, among many others. Machine learning on graphs, especially network representation learning, has shown remarkable performance in network-based applications, such as node/graph classification, graph clustering, and link prediction. Like performance, it is equally crucial for individuals to understand the behavior of machine learning models and be able to explain how these models arrive at a certain decision. Such needs have motivated many studies on interpretability in machine learning. For example, for social network analysis, we may need to know the reasons why certain users (or groups) are classified or clustered together by the machine learning models, or why a friend recommendation system considers some users similar so that they are recommended to connect with each other. Therefore, an interpretable network representation is necessary and it should carry the graph information to a level understandable by humans.

Here, we first introduce our method on interpretable network representations: the *network shape*. It provides a framework to represent a network with a 3-dimensional shape, and one can customize network shapes for their need, by choosing various graph sampling methods, 3D network embedding methods and shape-fitting methods. In this thesis, we introduce the two types of network shape: a *Kronecker hull* which represents a network as a 3D convex polyhedron using stochastic Kronecker graphs as the network embedding method, and a *Spectral Path* which represents a network as a 3D path connecting the spectral moments of the network and its subgraphs.

We demonstrate that network shapes can capture various properties of not only the network, but also its subgraphs. For instance, they can provide the distribution of subgraphs within a network, e.g., what proportion of subgraphs are structurally similar to the whole network? Network shapes are interpretable on different levels, so one can quickly under-

stand the structural properties of a network and its subgraphs by its network shape. Using experiments on real-world networks, we demonstrate that network shapes can be used in various applications, including (1) network visualization, the most intuitive way for users to understand a graph; (2) network categorization (e.g., is this a social or a biological network?); (3) computing similarity between two graphs. Moreover, we utilize network shapes to extend biometrics studies to network data, by solving two problems: *network identification* (Given an anonymized graph, can we identify the network from which it is collected? i.e., answering questions such as "where is this anonymized graph sampled from, Twitter or Facebook?") and *network authentication* (If one claims the graph is sampled from a certain network, can we verify this claim?). The overall objective of the thesis is to provide a compact, interpretable, visualizable, comparable and efficient representation of networks.

Interpretable Network Representations

By

Shengmin Jin

B.S., Fudan University, 2010

M.S., Syracuse University, 2016

Dissertation

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer/Information Science and Engineering

Syracuse University

December 2022

Acknowledgements

First, I would like to thank my advisor Dr. Reza Zafarani. When I started my PhD program, I knew little about how to do scientific research. It was Dr. Zafarani who taught me how to be a good researcher. He can quickly understand me and he always gives me excellent advice. He is not only a great advisor but also my best collaborator. I am lucky to be his first PhD student. Without his help, I would definitely not finish this long journey.

I must thank my mother Xiaofang Wu and my father Zhida Jin for their love and support. Because of the Cultural Revolution in China, they did not have the opportunity to go to college when they were young. However, they did their best to provide me with the best education. Making them proud is a huge motivation for me at all times.

I want to express my gratitude to Prof. Deepayan Chakrabarti, Prof. Biao Chen, Prof. Pinyuen Chen, and Prof. Vir Phoha for being my committee members. Their valuable comments and constructive suggestions have helped me significantly improve my thesis.

I also want to thank my labmates Reyhaneh Abdolazim, Jiayu Li, Hao Tian, and Xinyi Zhou. I will remember those days and nights working together. Thanks for giving me wonderful memories in the lab.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 General Introduction

Networks have become a universal language for describing complex data from science, engineering, and our daily life. Networks are used to study the role of a protein in biology [1], friendships in a social network [2], human emotions [3], among many other phenomena [4]. A compact, interpretable, visualizable, and efficient representation of networks facilitates scientific discoveries in a wide range of disciplines. Machine learning research aims to develop such network representations. Recent advances in network representation, e.g., in network embedding [5–7] or latent representation learning [8], aim to learn a mapping from a (sub)graph, or its nodes, to points in a low-dimensional vector space. For example, a three node graph such as $\triangle$ can be represented as a 2-dimensional vector: (1.24, 8.91). These techniques have shown remarkable performance in many applications, but face two fundamental limitations:

I. Interpretability. It is often difficult to understand the intuition behind learned representations. For instance, node (or subgraph) embedding techniques map nodes (or subgraphs) to points in a d-dimensional space, where no interpretation is often provided for such d dimensions. More specifically, when a graph is mapped to a point (a d-dimensional vector), one can hardly determine its exact structural properties from this vector, e.g., is it a dense network? The vector is mostly treated as a set of numerical features, limiting its usage.

II. Preserving Subgraph Information. As existing graph embedding approaches [9–11] map a network into a d-dimensional vector, the information on the subgraphs of this net-

work are mostly aggregated, or lost. Hence, given the embedding for the whole network, it is challenging to identify how embeddings for its subgraphs would look like. One might hypothesize that for a network with billions of nodes, samples (i.e., subgraphs) that are close in size to the original network should have similar embeddings; however, for a small subgraph such as a triad $\triangle$, which is a subgraph of many networks, the embedding should not be necessarily similar to that of the original network. Statistically speaking, graph embedding is taking a sample from a network (i.e., a subgraph) and computing a *statistic* (i.e., an embedding) for that one sample, ignoring the *sampling distribution*: the distribution of embedding values for all subgraphs. We denote the distribution of embedding values for all subgraphs of a network as the network's *embedding space*. With a graph representation that can provide (1) the network's embedding space, or (2) means to approximate the embedding of a subgraph, e.g., using the embeddings of the whole network and/or some of its other subgraphs, one can preserve subgraph information.

In descriptive statistics, a box plot visualization is often used to examine the distribution of data. From the box plot, one can quickly get the statistics of the data, such as minimum, first quartile, median, third quartile, and maximum values, and one can compare two datasets via their box plots. If there is something similar for networks, then one may answer the questions such as: What does the 25% or 75% of the network look like? What kind of structural properties do the network and its subgraphs have? Hence, in this work, we aim to propose a network representation which is visualizable, interpretable, comparable and preserving subgraph information.

1.2 Network Shapes

To address the aforementioned limitations, we propose to represent a network as a set of vectors, representing the network and its subgraphs. These vectors will represent the embedding space of the network. By ensuring that these vectors are in a 3-dimensional

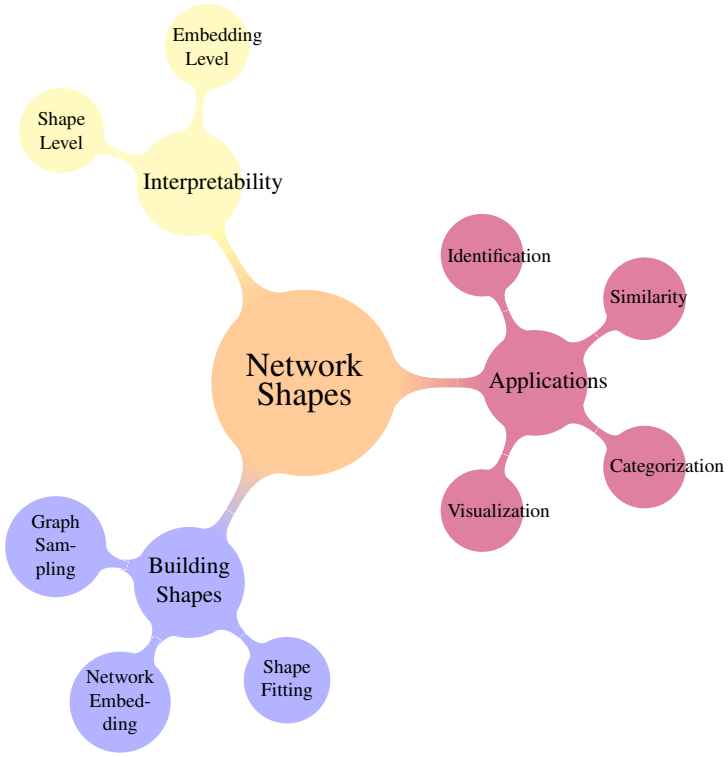
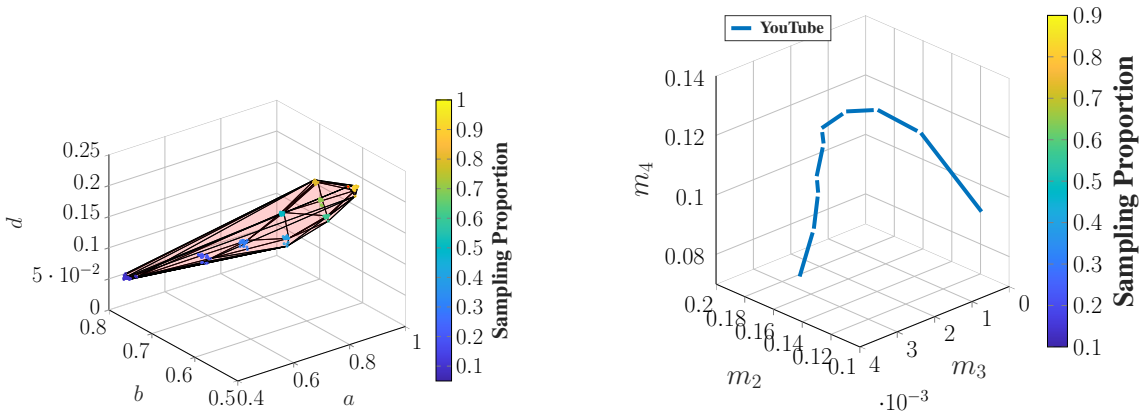


Figure 1.1: Organization of the Study

space, and by identifying a 3D shape that contains all such 3D vectors, the network (and its subgraphs) can be represented as a 3D shape. We denote this shape as the *network shape* of a network. As we will show, by representing networks with shapes, one can connect the structural properties of a network with the properties of its network shape (e.g., location, volume, patterns). For different networks, we may study their relationship through their network shapes (e.g., the overlap or distances between two 3D shapes).

In the thesis, we propose two types of network shapes: the *Kronecker Hull* and the *Spectral Path* (Examples are shown in Figure 1.2). In Figure 1.1, we demonstrate the organization of our study on network shapes, mainly from the following three aspects:

Building Network Shapes. We first introduce how to build a network shape. In general, it includes three steps: (a) to sample many subgraphs from the network, so we will discuss the graph sampling methods; (b) to map the network and sampled subgraphs to points in a 3D space, and we will introduce the 3D embedding methods used; (c) to fit a shape which contains the points in the space, and different shape fitting methods will be utilized. In



(a) Kronecker Hull for Hyves Social Network (b) Spectral Path of YouTube Social Network

Figure 1.2: Examples of Two Network Shapes

Table 1.1, we compare the two proposed network shapes in terms of their building steps.

Interpretability of Network Shapes. Then, we discuss how to interpret a network shape. We will investigate the interpretability of network shapes at both the embedding level and the shape level. On the embedding level, we aim to understand how to interpret each dimension of the embedding space, and on the shape level we want to study how the properties of network shapes (such as location, volume and shape patterns) are related to network structures and properties.

Applications of Network Shapes. Finally, we demonstrate the applications of network shapes, and they include network visualization, network categorization, graph similarity, network identification, and network authentication.

In the rest of the thesis, we introduce the first network shape: *Kronecker Hull* and we study its characteristics, interpretability and applications in Chapter 2; In Chapter 3, we propose extending biometrics studies to network data and we introduce two new problems in network studies: network identification and network authentication, and we demonstrate that network shapes can be used to solve them; In Chapter 4, we present a 3D spectral

Table 1.1: Two Types of Network Shapes

	Kronecker Hull	*Spectral Path*
Sampling	Random Node	Random Node
Embedding	Kronecker Points	Spectral Points
Fitting	3D convex hull	3D path

embedding method: *spectral points*, which utilizes the spectral moments of a graph to enhance the interpretability of a network shape as it is closely related to network structures and properties. We introduce the second network shape: *Spectral Path* in Chapter 5, and we discuss its interpretability and applications. Chapter 6 extends the study on spectral moments through its connection with network robustness. We discuss the future work of the study and conclude in Chapter 7. In the appendix, a few products on network shapes are introduced. More details are listed in Section 1.3.

1.3 Contribution of Dissertation

This research work discusses the interpretable network representations with 3D shapes on the methods and its applications. The contribution of this research are explained below.

1.3.1 Chapter 2: Network Shapes I: *Kronecker Hull*

We propose network shapes, a 3D representation for a network that (a) is easy to interpret; (b) captures various properties of not only the network, but also its subgraphs; (c) facilitates easy network visualization; and (d) enables various applications and comparative studies. We introduce the first network shape, a *Kronecker hull*, which represents a network as a 3D convex polyhedron. We study the properties of Kronecker hulls, such as the volume, the location, the internal points, and boundaries. The interpretability of Kronecker points and Kronecker hulls has also been investigated (e.g., how does a 10% subgraph of the network look like?). We demonstrate the utility of Kronecker hulls on applications such as network categorization (e.g., is this a social or a biological network?) and graph similarity.

1.3.2 Chapter 3: Network-based Biometrics

In this chapter, we extend biometrics studies to network data by formulating two new problems in network studies: network identification (i.e., identifying the source of an anonymized graph) and network authentication (i.e., verifying one's claim on the source of a graph). To solve the problems, we introduce the *network identity* and two identity types: *embedding-based identity*, and *shape-based identity* which utilize the network shapes. For network identification, we introduce two methods to predict the network from which a graph is sampled using the developed network identities. The first is a supervised learning method, which is highly accurate (84.4%). We also introduce an easier to implement method that relies on the distances between the sample embedding to the network identities, achieving a 70.8% accuracy. For network authentication problem, we propose two techniques: a *supervised splitter* with a low equal error rate, at which the false accept rate (FAR) is equal to the false reject rate (FRR); and a *Voronoi splitter*, which allows controlling the false reject with an acceptable false accept rate across networks.

1.3.3 Chapter 4: Embedding Network with Spectral Moments

In this chapter, we introduce *Spectral Point*, a 3D network embedding method that uses the truncated spectral moments of the network. Spectral points have the following advantages: (i) each dimension is closely related to the network structure and various network properties, so it is easy to interpret; (ii) the embedding space can help characterize various types of networks; (iii) the embedding space provides easy network visualization; and (iv) the embeddings are easy to compute. We demonstrate the interpretability of spectral points by mathematically proving their relationship to basic subgraphs such as triangles and squares, and the relationship to the network properties such as the degree distribution and the global clustering coefficient. We mathematically derive spectral moments for various types of graphs such as complete graphs, cycles, star graphs, and complete bipartite graphs. We

compute the spectral moments of real-world graphs from various categories and show that their structure and properties identified in past research is captured by spectral moments. We demonstrate that spectral moments can be used for network identification.

1.3.4 Chapter 5: Network Shapes II: *Spectral Path*

We propose the second network shape: the *Spectral Path*, which is a path connecting the spectral moments of the network and its subgraphs. We study the interpretability of spectral paths by investigating the *shapes* of spectral paths. We provide the theoretical relationship between the spectral moments of a network and those of its subgraphs. We illustrate how this relationship is closely related to the network structure. We show that spectral path can be used for applications such as network visualization and network identification. We theoretically explore the possibility of using the expected spectral moments of subgraphs to help distinguish cospectral graphs.

1.3.5 Chapter 6: Spectral Moments and Network Robustness

We find that the second spectral moment m_2 of a network can be used as a network robustness measure. We show that m_2 captures network robustness on both synthetic and real-world networks. Specifically, when m_2 is smaller, the network is more robust. The spectral moments can be used to assess the degree of robustness of a network, or to compare the robustness of two networks varying in size. We show that we can control the network robustness by manipulating its m_2 value, to design a network that is more robust under failures. We conduct experiments on real-world networks, and evaluate the method. We demonstrate that with m_2 as the robustness measure, one can study how a complex networked system behaves under cascading failures by looking at how network robustness evolves. By studying cascading failures in a power grid network, we show that after an initial failure making the grid vulnerable, the grid stabilizes after the cascading failures.

1.3.6 Appendix A: Products of Network Shapes

Here, we introduce a web platform WEBSHAPES that enables researchers and practitioners to visualize their network data as customized network shapes. We also provide a repository of precomputed network shapes for various networks, which includes 102 public networks from 12 different categories.

1.4 Publications

Part of the works presented in this thesis has appeared in multiple publications:

Conferences:

1. Jin, Shengmin, and Reza Zafarani. "Representing networks with 3d shapes." In 2018 IEEE International Conference on Data Mining (ICDM), IEEE, 2018 [12].
2. Jin, Shengmin, Vir Phoha, and Reza Zafarani."Network identification and authentication." 2019 IEEE International Conference on Data Mining (ICDM). IEEE, 2019 [13].
3. Jin, Shengmin, Richard Wituszynski, Max Caiello-Gingold, and Reza Zafarani. "WebShapes: Network Visualization with 3D Shapes." Proceedings of the 13th International Conference on Web Search and Data Mining. 2020 [14].
4. Jin, Shengmin, and Reza Zafarani."The spectral zoo of networks: Embedding and visualizing networks with spectral moments." Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2020 [15].
5. Abdolazimi, Reyhaneh, Shengmin Jin, and Reza Zafarani. "Noise-enhanced community detection." Proceedings of the 31st ACM Conference on Hypertext and Social Media. 2020 [16].
6. Jin, Shengmin, Hao Tian, Jiayu Li, and Reza Zafarani."A Spectral Representation of

Networks: The Path of Subgraphs." Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022 [17].

7. Jin, Shengmin, Rui Ma, Jiayu Li, Sara Eftekharnejad, and Reza Zafarani. "A Spectral Measure for Network Robustness: Assessment, Design, and Evolution." 2022 IEEE International Conference on Knowledge Graph. IEEE, 2022 [18].

Journals:

1. Jin, Shengmin, Vir Phoha, and Reza Zafarani. "Graph-based Identification and Authentication: A Stochastic Kronecker Approach." IEEE Transactions on Knowledge and Data Engineering (2020) [19].

Tutorials:

1. Jin, Shengmin, Danai Koutra, and Reza Zafarani. "Interpretable Network Representations."[1] In Companion Proceedings of the Web Conference 2022 [20].

Moreover, some of our publications are not included in this thesis:

Conferences:

1. Jin, Shengmin, and Reza Zafarani. "Emotions in social networks: Distributions, patterns, and models." Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. 2017 [3].

2. Jin, Shengmin, and Reza Zafarani. "Sentiment Prediction in Social Networks." 2018 IEEE International Conference on Data Mining Workshops (ICDMW) [21].

3. Zhou, Xinyi, Shengmin Jin, and Reza Zafarani. "Sentiment Paradoxes in Social Networks: Why Your Friends Are More Positive Than You?." Proceedings of the International AAAI Conference on Web and Social Media. Vol. 14. 2020 [22].

4. Ma, Rui, Shengmin Jin, Sara Eftekharnejad, Reza Zafarani, and Wolf Peter Jean

[1] `https://shengminjin.github.io/tutorials/www2022`

Philippe. "A probabilistic cascading failure model for dynamic operating conditions." IEEE Access 8 (2020): 61741-61753 [23].

5. Li, Jiayu, Tianyun Zhang, Hao Tian, Shengmin Jin, Makan Fardad, and Reza Zafarani. "Sgcn: A graph sparsifier based on graph convolutional networks." In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 275-287. Springer, Cham, 2020 [24].

6. Li, Jiayu, Tianyun Zhang, Shengmin Jin, Makan Fardad, and Reza Zafarani. "AdverSparse: An Adversarial Attack Framework for Deep Spatial-Temporal Graph Neural Networks." In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5857-5861. IEEE, 2022 [25].

Journals:

1. Li, Jiayu, Tianyun Zhang, Hao Tian, Shengmin Jin, Makan Fardad, and Reza Zafarani. "Graph sparsification with graph convolutional networks." International Journal of Data Science and Analytics 13, no. 1 (2022): 33-46 [26].

1.5 Related Work

Overall, our research is mainly related to interpretable network representation techniques from the following three areas:

Network embedding methods. Network embedding techniques aim to map a node (or a graph) into a low-dimensional vector and efficiently preserve the network structure. Some of them are interpretable such as those that embed nodes based on return probabilities of random walks [27], or those that perform graph embedding using spectral signatures (e.g., network density of states (DOS) [28, 29], trace signature based on heat diffusion [30]. In this work, we introduce two network embeddings for the use of network shapes: *Kronecker Points* and *Spectral Points*. Compared to other embeddings, Kronecker Points and Spectral

Points capture network structures and network properties with three values, making the embeddings visualizable.

Graph summarization methods. Graph summarization aims to find a short summary of large graphs to capture their characteristics, and these methods often provide an aggregated summary of a network [31]. In general, the output of graph summarization methods is either a "supergraph", a sparsified graph, or a structure list [31]. For example, grouping-based graph summarization use clustering or community detection algorithms to group nodes into "supernodes", so that the original network is summarized in terms of a smaller graph (a supergraph) with several supernodes. Another example is VoG [32], or *vocabulary-based summarization of graphs*, which constructs a 'vocabulary' of subgraphs with network motifs and finds the most succinct description of a graph in terms of the vocabulary. Both examples investigate the relationship between a network and its subgraphs. One can also view network shapes a special kind of graph summarization method, which summarize graphs with shapes in the 3D embedding space.

Network visualization. Visual displays of networks often can lead to a better understanding of networks. Examples include graph layout studies (e.g., spectral graph drawing [33, 34]) that aim to draw graphs (node-edge diagrams) in an aesthetically-pleasing way. Moreover, network visualization methods can also be used to enhance the interpretability of network embedding and graph summarization methods. Network shapes provide an interpretable and visualizable representation of networks.

Reconstruction Conjecture. Bollobás has shown that almost all graphs are reconstructible [35]. Moreover, not all subgraphs are necessary to reconstruct them: for almost all graphs, there exist three subgraphs that uniquely identify the graph. Our setting matches that of the theoretical findings of Bollobás. Specifically, by taking samples from the graph for creating network shapes, we aim to capture the subgraphs that can uniquely represent a graph.

Chapter 2

Network Shapes I: *Kronecker hull*

2.1 Introduction

There has been a surge of interest in machine learning in graphs, as graphs and networks are ubiquitous across the globe and within science and engineering: road networks, power grids, protein-protein interaction networks, scientific collaboration networks, social networks, to name a few. Recent machine learning research has focused on efficient and effective ways to represent graph structure. Existing graph representation methods such as network embedding techniques learn to map a node (or a graph) to a vector in a low-dimensional vector space. However, the mapped values are often difficult to interpret, lacking information on the structure of the network or its subgraphs. Instead of using a low-dimensional vector to represent a graph, we propose to represent a network with a 3-dimensional shape: the *network shape*. We introduce the first network shape, a *Kronecker hull*, which represents a network as a 3D convex polyhedron using stochastic Kronecker graphs. We present a linear time algorithm to build Kronecker hulls. Network shapes provide a compact representation of networks that is easy to visualize and interpret. They captures various properties of not only the network, but also its subgraphs. For instance, they can provide the distribution of subgraphs within a network, e.g., what proportion of subgraphs are structurally similar to the whole network? Overall, the contributions are mainly the following:

1. We propose network shapes, a 3D representation for a network that (i) is easy to interpret; (ii) captures various properties of not only the network, but also its sub-

graphs; (iii) facilitates easy network visualization; and (iv) enables various applications and comparative studies;

2. We propose Kronecker Hull, a network shape that represents a network and its subgraphs via a convex polyhedron in the three dimensional space;

3. We demonstrate how properties of a Kronecker hull (e.g., its volume or location) are connected to the structure of the network it represents. We study Kronecker hull properties using extensive experiments on eighteen real-world networks from four different categories;

4. We show the interpretability of Kronecker hull, which helps to characterize graphs (e.g., how does a 10% subgraph look like?); and

5. We show applications of network shapes in network categorization (e.g., is this a social or a biological network?), and computing graph similarity.

Implications of Network Shapes. Representing networks as 3D shapes has multiple benefits and applications:

► *Compact Representation of Networks*. Network shapes can help represent networks (and their embedding space) compactly. In most of our experiments, we can represent networks with million of nodes using shapes that can be represented with less than 40 boundary points.

► *Visualizing Networks*. Visualizing large graphs is challenging. This difficulty lies in the natural clutter, crossing, and overdrawing issues [36]. Network shapes help visualize networks (and their embedding space) with limited clutter.

► *Interpretation.* By properly designing network shapes, they can help illustrate structural properties of graphs and how a network is composed of subgraphs with different properties.

► *Features.* Features from network shapes such as their boundary points, center of gravity, volume, and other geometrical properties can capture various information about the network and its subgraphs and can be used for machine learning.

The rest of the chapter is organized as follows. In Section 2.2, we detail the necessary steps to build network shapes. In Section 2.3, we discuss stochastic Kronecker graphs, the foundation behind Kronecker hulls (a network shape). Section 2.4 provides the algorithm for computing the Kronecker Hull and its time complexity analysis. We summarize our experimental setup and data in Section 2.5. With various experiments, we look into the characteristics of Kronecker Hulls including their volume, location, internal points and boundries in Section 2.6. In Section 2.7, we discuss the interpretability of Kronecker Hulls. Section 2.8 provides some applications which utilize network shapes. After reviewing additional related work in Section 2.9, we conclude the chapter in Section 2.10.

2.2 Building Network Shapes

The following simple steps can help build a network shape:

Step 1: **Sample** many subgraphs from the network
Step 2: **Map** the network and its subgraphs to 3D vectors
Step 3: **Fit** a 3D Shape to the set of 3D vectors

The **first** requirement for constructing network shapes is a *sampling method*. Any sampling method can work. In our algorithm, we have utilized *Random Node Sampling* strategy [37]. Random node sampling selects a proportion p of nodes from a graph uniformly at random and the sample subgraph is then the graph induced by these selected nodes. Random node sampling is shown to perform well for various network measurements [37]

and is a fast algorithm with linear time complexity. To sample systematically, one can sample by varying proportions of nodes (e.g., from 0% to 100%) with some fixed step size s. To control variations, for each proportion, one can sample t independently sampled subgraphs, i.e., a total of $t \times s$ subgraphs for one network.

The **second** requirement for constructing network shapes is an *embedding technique that can map a network to a 3D point*. The technique should provide embedding vectors that are easy to interpret and can capture the properties of the network and its subgraphs. Given such a technique, we can represent a network and its subgraphs as a set of 3D points. Similarly, one can think of many fast techniques to map a graph into a 3D vector, e.g., represent it with its (*diameter*, *average path length*, *clustering coefficient*). Here, we consider Stochastic Kronecker Graphs [38] as an appropriate candidate for mapping a graph into an interpretable 3D point, which we denote as the *Kronecker point*. In Section 2.3, we investigate the properties and interpretation of Kronecker points.

The **third** and final requirement for building network shapes is a *technique to fit a 3D shape to a set of 3D points* obtained in Step 2 (3D embedding). While this can be done by fitting a variety of shapes (e.g., spheres), we consider building a network shape from a set of 3D points by computing its convex hull. A convex hull, for a set of points in a Euclidean space, is the smallest convex set that contains all the points in the original set [39]. Convex hull of a finite set of n points in a three-dimensional space can be computed with at most $\mathcal{O}(n \log n)$ operations [40].

2.3 Stochastic Kronecker Graphs

Stochastic Kronecker graphs [38] provide an approach to model large-scale graphs using the *Kronecker product* $\otimes$ matrix operation. The Kronecker product generalizes matrix

outer product, e.g., the Kronecker product of $\left[\begin{smallmatrix}1&2\\3&4\end{smallmatrix}\right]$ and $\left[\begin{smallmatrix}0&5\\6&7\end{smallmatrix}\right]$, denoted as $\left[\begin{smallmatrix}1&2\\3&4\end{smallmatrix}\right] \otimes \left[\begin{smallmatrix}0&5\\6&7\end{smallmatrix}\right]$ is

$$\left[\begin{smallmatrix}1&2\\3&4\end{smallmatrix}\right] \otimes \left[\begin{smallmatrix}0&5\\6&7\end{smallmatrix}\right] = \begin{bmatrix} 1\cdot\left[\begin{smallmatrix}0&5\\6&7\end{smallmatrix}\right] & 2\cdot\left[\begin{smallmatrix}0&5\\6&7\end{smallmatrix}\right] \\ 3\cdot\left[\begin{smallmatrix}0&5\\6&7\end{smallmatrix}\right] & 4\cdot\left[\begin{smallmatrix}0&5\\6&7\end{smallmatrix}\right] \end{bmatrix} = \begin{bmatrix} 1\cdot0 & 1\cdot5 & 2\cdot0 & 2\cdot5 \\ 1\cdot6 & 1\cdot7 & 2\cdot6 & 2\cdot7 \\ 3\cdot0 & 3\cdot5 & 4\cdot0 & 4\cdot5 \\ 3\cdot6 & 3\cdot7 & 4\cdot6 & 4\cdot7 \end{bmatrix}$$
$$= \begin{bmatrix} 0 & 5 & 0 & 10 \\ 6 & 7 & 12 & 14 \\ 0 & 15 & 0 & 20 \\ 18 & 21 & 24 & 28 \end{bmatrix}.$$

When modeling a network using Stochastic Kronecker graphs, we aim to learn a small probability matrix $P \in \mathbb{R}^{n\times n}$, known as the *Kronecker initiator matrix*, such that the k^{th} Kronecker power of P (i.e., $P^{\otimes k} = \underbrace{P \otimes P \cdots \otimes P}_{k \text{ times}}$) is most likely to have generated the adjacency matrix $A \in \mathbb{R}^{n^k \times n^k}$ of the graphs which we are modeling, i.e., $P(A|P)$ is maximized (for further details refer to Ref. [38]). The KRONFIT algorithm can estimate the Kronecker initiator matrix for a real-world graph using maximum likelihood and in linear time [38].

2.3.1 Kronecker Points

Consider fitting a 2×2 Kronecker initiator matrix $I = \left[\begin{smallmatrix}a&b\\c&d\end{smallmatrix}\right]$ to a network. In an undirected network, where the adjacency matrix is symmetric, the Kronecker initiator matrix learned is also symmetric, i.e., $b = c$. Thus, one can embed an undirected network, or any of its subgraphs, to a point (a, b, d) in the 3-D space, which we denote as the *Kronecker point* of a graph. As a 3D point, Kronecker point can be easily visualized.

Kronecker points (a, b, d) have basic properties:

I. By definition, Kronecker initiator matrices are probability matrices, i.e., values a, b and d are all between 0 and 1. Hence, all possible graphs can be embedded in a $1 \times 1 \times 1$ cube.

II. Without loss of generality, we can assume that $a \geq d$. Consider two initiator matrices $\left[\begin{smallmatrix}a&b\\b&d\end{smallmatrix}\right]$ and $\left[\begin{smallmatrix}d&b\\b&a\end{smallmatrix}\right]$ that only differ with respect to the positions of a and d, i.e., we can obtain one by simultaneous shuffling of rows and columns of the other based on some

permutation. Calculating the k^{th} Kronecker power of both initiator matrices yields two adjacency matrices for two graphs. We can simply prove that these two graphs are indeed the same graph, i.e., the graphs are *isomorphic*. Assume P is a permutation matrix: a square binary matrix with exactly one entry of 1 in each row and column, and 0s elsewhere. Let X denote any initiator matrix. Then PXP^T represents a simultaneous shuffling of rows and columns of X according to permutation P. By Kronecker product properties $(PXP^T)^{\otimes k} = P^{\otimes k}X^{\otimes k}(P^{\otimes k})^T$. As $P^{\otimes k}$ is also a permutation matrix, the graph represented by adjacency $(PXP^T)^{\otimes k}$ is the same as the one by $X^{\otimes k}$.

2.3.2 KRONFIT Limitations

KRONFIT can provide Kronecker points, but has a few limitations that may lead to over/underestimation. When the number of nodes within a real-world network is not a power of 2, KRONFIT will add isolated nodes so that the number of nodes becomes a power of 2 [41]. Adding isolated nodes may lead to underestimation of the parameters as it decreases the overall edge density and core strength of the groups. On the other hand, as the input to KRONFIT is a list of edges, when the network is extremely sparse and the graph size is small, KRONFIT can overestimate as it overlooks real isolated nodes within the network. In Section 2.7.1, the overestimation in sparse random network fits this second case.

2.4 Kronecker Hull

We introduce an algorithm to obtain the Kronecker hull of a network, and analyze its time complexity. The algorithm pseudocode is provided in Algorithm 1. The algorithm utilizes *Random Node Sampling* to sample many subgraphs from the network by (1) varying the proportion of nodes from 0% to 100% with step size s and (2) taking t independent samples for each proportion. For each sample (and the whole network), the algorithm computes its Kronecker point via KRONFIT algorithm. Finally, the convex hull of these Kronecker

Algorithm 1 KRONECKER HULL algorithm

input : an undirected network graph: $G(V, E)$
output : the Kronecker hull of G: KH_G
parameter : s : sampling proportion step size;
t : number of samples for one proportion;
Kronecker_points = { };

for ($p = s;\ p < 100\%;\ p = p + s$) {
  for ($i = 1;\ i \leq t;\ i = i + 1$) {
    %Sample a subgraph G_p
    G_p = RandomNodeSampling(G, p);
    %Fit Kronecker Initiator to G_p $\left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$ = KRONFIT($G_s, 2$);
    Kronecker_point = (a, b, d);
    Kronecker_points.add(Kronecker_point);
  }
}
$\left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$ = KRONFIT($G, 2$);
Kronecker_point = (a, b, d);
Kronecker_points.add(Kronecker_point);
KH_G = Quickhull(Kronecker_points);%Convex Hull
return KH_G;

points are computed, using Quickhull algorithm [42], to obtain the Kronecker hull of the graph. The implementation is available at: `https://github.com/shengminjin/KroneckerHull`

Time Complexity. For one subgraph, random node sampling takes $\mathcal{O}(n + m)$ and KRONFIT takes $\mathcal{O}(n + m)$, where $|V| = n$ and $|E| = m$. Hence, for each subgraph, the time complexity is $\mathcal{O}(n + m)$. We have a total of $\frac{100}{s} \times t + 1$ graphs (a network and its subgraphs) for which we compute Kronecker points. As the number of Kronecker points is very small compared to the size of the network, the time spent on computing the convex hull is constant. Hence, the time complexity to compute Kronecker hull is $\mathcal{O}(\frac{t}{s}(n + m))$, linear in the number of nodes and edges.

2.5 Experimental Setup

For our experiments, we generate Kronecker hulls for various real-world networks by varying the proportion of nodes from 0% to 100% with step size 10%, i.e., $s = 10\%$ in Algorithm 1; for each proportion (except for 100% which represents the whole graph), we generate 20 independently sampled subgraphs, i.e., $t = 20$ in Algorithm 1. In total, we generate $20 \times 9 + 1 = 181$ Kronecker points for each network, using which we obtain the Kronecker hull for the network. Next, we summarize the network data used in our experiments.

2.5.1 Datasets

For our experiments, we use eighteen real-world networks from four general network categories: social networks, collaboration networks, road networks, and biological networks.

Social Networks: In total, we have eight social networks from three sub-categories.

► Location-based Social Networks:

Brightkite and *Gowalla* [43]: were both once location-based social networking sites where users shared their locations by checking-in. Both networks were originally directed but have been converted to undirected where an undirected edge between users exist when friendships in both directions exist.

► Friendship-based Social Networks:

Hyves [44]: the most popular social networking site in the Netherlands with mainly Dutch visitors. It competes with sites such as Facebook and MySpace in that country.

Orkut [43]: was a social networking website owned and operated by Google, shutdown in 2014.

Livejournal [45]: a social network where users can keep a blog or journal. Users can form friendship or follow others. In this dataset, edges represent friendships (undirected).

MySpace [45]: a social network with emphasis on music.

► **Video-Sharing or Movie Sites:**

YouTube [43]: a video-sharing site with a social network.

Flixster [44]: a social movie site allowing users to buy, rent, or watch movies, share ratings, and discover new movies.

Collaboration Networks: We include four collaboration networks from arXiv.org, which include scientific collaborations between authors with different scientific interests. In a collaboration networks, an undirected edge between nodes i and j exists, if authors i and j have co-authored at least one paper.

Astro-Ph [43]: Astro physics.

Cond-Mat [43]: Condense matter physics.

Gr-Qc [43]: General relativity and quantum cosmology.

Hep-Th [43]: High energy physics theory.

Road Networks: We include three road networks. In road networks, nodes are intersections/endpoints and undirected edges are the roads connecting these intersections/road endpoints.

Road-CA [43]: the road network of California.

Road-PA [43]: the road network of Pennsylvania.

Road-TX [43]: the road network of Texas.

Biological Networks: We include three biological networks.

Table 2.1: Dataset Statistics

Type	Network	$\|V\|= n$	$\|E\|= m$	Average Degree	Clustering Coefficient
Social Networks	Brightkite	58,228	214,078	7.353	0.1723
	Flixster	2,523,386	7,918,801	6.276	0.0834
	Gowalla	196,591	950,327	9.668	0.2367
	Hyves	1,402,673	2,777,419	3.960	0.0448
	Livejournal	3,017,286	85,654,976	56.776	0.1196
	MySpace	854,498	5,635,296	13.190	0.0433
	Orkut	3,072,441	117,185,083	76.281	0.1666
	YouTube	1,134,890	2,987,624	5.265	0.0808
Collaboration Networks	Astro-Ph	18,772	198,050	21.100	0.6306
	Cond-Mat	23,133	93,439	8.078	0.6334
	Gr-Qc	5,242	14,484	5.526	0.5296
	Hep-Th	9,877	25,973	5.259	0.4714
Road Networks	Road-CA	1,965,206	2,766,607	2.816	0.0464
	Road-PA	1,088,092	1,541,898	2.834	0.0465
	Road-TX	1,379,917	1,921,660	2.785	0.0470
Biological Networks	Bio-Dmela	7,393	25,569	6.917	0.0119
	Bio-Grid-Yeast	5,870	313,890	104	0.0516
	Human-Brain	177,600	15,669,036	176	0.4580

Bio-Grid-Yeast [43]: a protein-protein interaction networks.

Bio-Dmela [43]: a protein-protein interaction networks.

Human-Brain [43]: the network of human brain.

The data statistics are summarized in Table 2.1. To assess the impact of network structure on Kronecker hulls, for each real-world network, we generate a random synthetic network with a perturbed network structure, but with the same degree distribution, i.e., a *null model*. We create the null model using the configuration model [46], which can generate a random network with the same degree distribution and edge density (i.e., $|E|/\binom{|V|}{2}$) as the given real-world network.

2.6 Kronecker Hull Characteristics

To investigate the characteristics of Kronecker Hulls, we compute the Kronecker Hull for all networks. Figure 2.1 provides the Kronecker hull for one of our social networks: Hyves. The points on the boundary (or within) the Kronecker hull are Kronecker points (a, b, d) representing different sampling proportions. The Kronecker points are colored

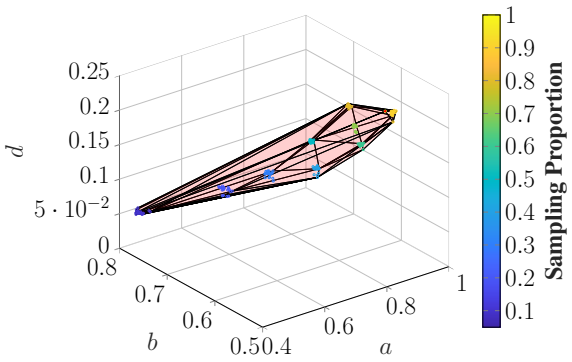


Figure 2.1: Kronecker Hull for Hyves Social Network

differently for different sampling proportions. We investigate different characteristics of Kronecker Hulls, but more importantly how the structure of a network is connected to those characteristics. In particular, we look at the volume, location, internal points, and boundaries of Kronecker hulls.

2.6.1 Volume of Kronecker Hulls

As a Kronecker hull is a convex hull, its volume can be easily computed via triangulation. How is the volume of a Kronecker hull connected to the properties of the network it represents? Table 2.2 provides the volumes of the Kronecker hulls, denoted by $\mathsf{volume}(G)$, for all networks. We observe that for social, road, and biological networks, volumes are between 3.5×10^{-5} to 1.7×10^{-3}. The maximum possible volume of a Kronecker hull can be 1 as values a, b, and d lie in range $[0, 1]$. Hence, the Kronecker hulls of these networks are compact from a volume perspective, taking up only about one thousandth of the whole space. Volumes of collaboration networks are much larger, varying from 3.4×10^{-3} to 0.2, which we speculate is due to their specific network structure.

To investigate the impact of network structure on the volume of Kronecker hull, for any graph G, we compare the volume of its Kronecker hull $\mathsf{volume}(G)$ to that of its null model $\mathsf{volume}(G_{null})$. Note that null models have the same edge density and degree distribution as the original graph, but with a random network structure. Hence, any change in volume indicates that network structure has an impact on volume. To compare volumes, we com-

Table 2.2: Kronecker Hull Volume

Type	Networks	Edge Density ($\times 10^{-4}$)	Actual Graph		Null-Model		Ratio $\frac{\mathsf{volume}(G)}{\mathsf{volume}(G_{null})}$	$\mathsf{overlap}(G, G_{null})$
			$\mathsf{volume}(G)$ ($\times 10^{-4}$)	Clustering Coefficient	$\mathsf{volume}(G_{null})$ ($\times 10^{-4}$)	Clustering Coefficient		
Social Networks	Brightkite	1.262	6.22	0.1723	8.56	0.0053	0.73	70.49%
	Flixster	0.025	3.42	0.0834	4.39	0.0012	0.78	83.87%
	Gowalla	0.492	13.00	0.2367	7.86	0.0103	1.61	51.63%
	Hyves	0.028	4.22	0.0448	5.52	0.0030	0.76	78.44%
	Livejournal	0.188	1.93	0.1196	1.44	0.0013	1.34	75.69%
	MySpace	0.154	3.75	0.0433	2.90	0.0037	1.29	86.37%
	Orkut	0.248	0.44	0.1666	0.99	0.0006	0.45	26.38%
	YouTube	0.046	5.87	0.0808	5.94	0.0065	0.99	90.86%
Collaboration Networks	Astro-Ph	11.241	34.00	0.6306	3.89	0.0094	8.67	0%
	Cond-Mat	3.492	98.00	0.6334	15.00	0.0022	6.36	16.99%
	Gr-Qc	10.544	200.00	0.5296	15.00	0.0053	13.50	8.23%
	Hep-Th	5.325	90.00	0.4714	13.00	0.0018	7.19	29.07%
Road Networks	Road-CA	0.014	10.00	0.0464	11.00	3.4×10^{-7}	0.89	87.57%
	Road-PA	0.026	9.99	0.0465	9.72	3.5×10^{-6}	1.03	72.89%
	Road-TX	0.020	7.05	0.0470	8.31	0.0000	0.85	74.98%
Biological Networks	Bio-Dmela	9.360	17.00	0.0119	20.00	0.0067	0.85	95.35%
	Bio-Grid-Yeast	173.950	9.72	0.0516	10.00	0.0694	0.97	91.72%
	Human-Brain	9.937	0.35	0.4580	0.10	0.0169	3.50	33.59%

pute the ratio $\frac{\mathsf{volume}(G)}{\mathsf{volume}(G_{null})}$. While we observe that for all networks, the ratio is not equal to 1, indicating that network structure has an impact on the volume, the ratio often takes a value between 0.5 to 2, i.e., the actual volume can be at most twice, or at least half of that of its null model. We believe this finding can have implications in finding proper null models. As speculated, collaboration networks are outliers, with their network structures most damaged when constructing their null models: their Kronecker hull volume is much larger than that of their null models. Our further analysis indicated a strong correlation ($\rho = 0.88$) between volume ratios and the clustering coefficient of networks, which is high in collaboration networks and is dramatically reduced in null models. We also conducted a multiple linear regression to predict volume based on five predictors: $|V|$, $|E|$, edge density, average degree, and clustering coefficient. The regression coefficients also indicated that volume is strongly correlated to the edge density and clustering coefficient, with regression coefficients being nearly 0 for the other three variables.

2.6.2 Location of Kronecker Hulls

To identify network properties that impact the location of a Kronecker hull, one must seek properties that when changed within a network, the new Kronecker hull for the modified network is at a different location in the 3D space, i.e., has less than 100% overlap with

the original Kronecker hull. Hence, to investigate the impact of network structure on Kronecker hull location, we compute the overlap between Kronecker hulls of networks with that of their null models. We define the overlap between Kronecker hulls for networks A and B as

$$\mathsf{overlap}(A, B) = \frac{\mathsf{volume}(\mathsf{KH}_A \cap \mathsf{KH}_B)}{\min(\mathsf{volume}(\mathsf{KH}_A), \mathsf{volume}(\mathsf{KH}_B))}\ , \tag{2.1}$$

where volume is the volume of a Kronecker hull, and KH_A and KH_B represent Kronecker hulls of graphs A and B, respectively. We define the overlap as a ratio: the volume of the intersection $\mathsf{KH}_A \cap \mathsf{KH}_B$ normalized by the volume of the smaller Kronecker hull. It is easy to prove that given any collection of convex sets (finite, countable or uncountable), their intersection is a convex set. Therefore, the intersection of two Kronecker hulls $\mathsf{KH}_A \cap \mathsf{KH}_B$ is also convex, allowing us to easily compute its volume. The results are in Table 2.2. We observe an overlap that is less than 100% in all networks, indicating that network structure has an impact on the location of Kronecker hulls. Similar to our observations with respect to volume, (i) collaboration networks are outliers with very small overlaps and (ii) clustering coefficients of networks are strongly negatively correlated ($\rho = -0.86$) to their overlaps. In addition, ratios $\frac{\mathsf{volume}(G)}{\mathsf{volume}(G_{null})}$ are strongly negatively correlated ($\rho = -0.77$) to overlaps $\mathsf{overlap}(G, G_{null})$ indicating that, e.g., when network structure is damaged, Kronecker hulls shrink in volume and move far from their original location.

2.6.3 Internal Points

By definition, a point within a Kronecker hull of a network represents a sample from this network, i.e., a subgraph. Here, we investigate (1) how samples are distributed within a Kronecker hull and (2) how distances between samples are connected to similarities between corresponding subgraphs.

► *Sample Distribution*. In the Hyves example provided in Figure 2.1, a clustering phenomenon is observed: points representing samples of the same proportion appear to be

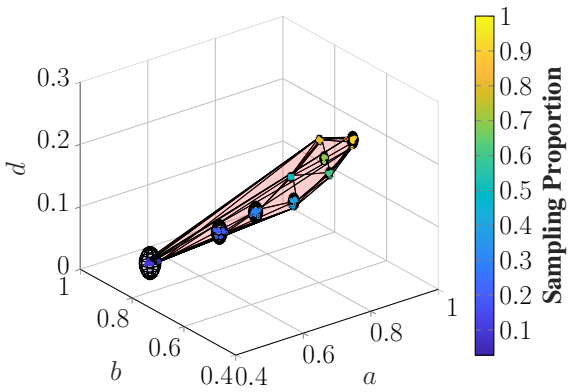
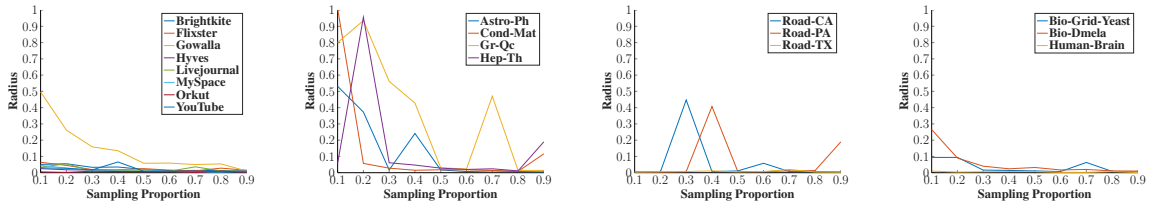


Figure 2.2: Kronecker Hull of Hyves with Sphere fit

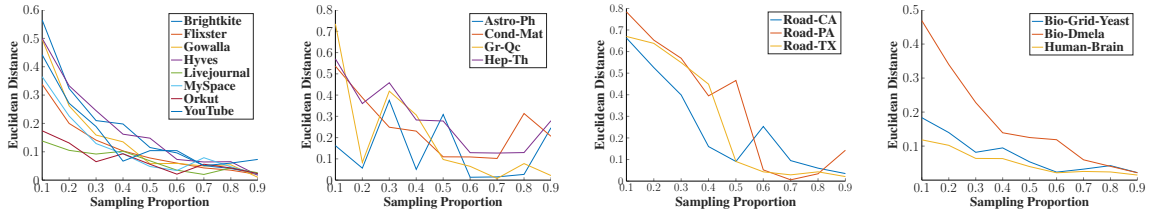


(a) Social Networks (b) Collab. Networks (c) Road Networks (d) Bio. Networks

Figure 2.3: Radius of Spheres Fit to Subgraph Kronecker Points

clustered. To verify whether such a clustering exists, we fit a sphere to the points that represent the same sample size (see Figure 2.2). The sphere better visualizes the location of the cluster and its radius captures the variance. For all networks, we compute the radii of all such spheres; the results are in Figure 2.3. We find that the clustering phenomenon is observed for most networks, with relatively small radii that decreases as the sampling proportions increase. Compared to other networks, the radii of spheres of collaboration networks are larger, especially in smaller samples, i.e., clustering is not obvious. We speculate that this observation is due to samples being taken from different academic communities within the graph. Overall, our observations indicate that given a point within a Kronecker hull, nearby points are likely to be samples of the same size.

► *Between-Sample Distances*. Consider two subgraphs of a network, each represented as a Kronecker point within the Kronecker hull of the network. Ideally, we hope that the distance between these two Kronecker points is related to the similarity between these two subgraphs. However, measuring similarity between two graphs can be challenging and subjective. To circumvent the challenge of computing graph similarity, we compute



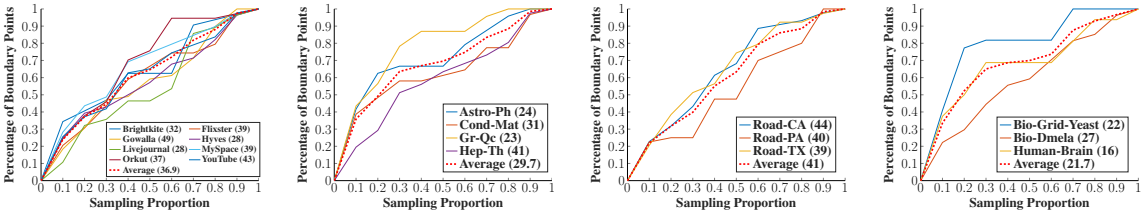
(a) Social Networks (b) Collab. Networks (c) Road Networks (d) Bio. Networks

Figure 2.4: Distances between Sphere Centers (representing subgraph Kronecker points) and the Whole-graph Kronecker Point

the distances between Kronecker points of graphs for which we have an intuitive understanding of their similarity. Here, we compute the distances between Kronecker points of different subgraphs and that of the whole network. This decision is based on the intuition that by increasing the sampling proportion, subgraphs should become more similar to the whole network (a 100% subgraph). As samples of the same proportion are clustered, we compute the Euclidean distance between the Kronecker point of the whole network and the sphere centers (representing Kronecker points for different sampling proportions). Figure 2.4 illustrates that with the increase in sampling proportion, sphere centers become closer to the Kronecker point of the whole network, indicating a convergence in Kronecker points as graphs become more similar. Looking at networks from different categories, we observe that (1) for social and biological networks, the distances drop fast when the sampling proportion increases from 10% to 30%, which suggests network structure of a 30% subgraph can be close to that of the whole network, when sampled using random node sampling; (2) for road networks, the sphere centers are far when the sampling proportion is small. With the increase in sampling proportion, the distances drop sharply when samples are below 60% and become very small after they reach 70%; (3) for collaboration networks, we observe a decreasing trend in distances, but unlike other networks, there is an oscillation.

Figure 2.5: Kronecker Hull Boundary Points Distribution. The numbers in the legend specify the number of boundary points.

(a) Social Networks (b) Collab. Networks (c) Road Networks (d) Bio. Networks



2.6.4 Boundaries

Figure 2.5 provides the number (in the legend) and the distribution of boundary points (vertices) of Kronecker hulls. We find that number of boundary points ranges from 16 to 49, and for most networks is between 30-40, out of the total of 181 points. Points from very small samples, especially those for sampling proportion 10%, are more likely to be boundary points. Points from middle size samples are more likely to be within the hull. Overall, we observe a continuity in points being on the boundary with the increase in sampling proportion. These findings suggest that (1) a limited number of points (e.g., 40) is required to store a Kronecker hull; (2) we can sample fewer points for each proportion to construct a Kronecker hull; and (3) boundary points can be used as compact features for machine learning on graphs.

2.7 Interpretability of *Kronecker Hulls*

2.7.1 Interpretability of Kronecker Points

One can interpret the 2×2 initiator $\left[\begin{smallmatrix} a & b \\ b & d \end{smallmatrix}\right]$ of an undirected network as a recursive expansion of two groups of network nodes into subgroups [38]. We can interpret a and d as the proportion of edges within each of the groups, and b as the proportion of edges between the two groups. As proved, we can assume that $a \geq d$; hence, we can split the whole space into three regions, i.e., split all possible networks into three types. Each region represents

a different network structure. We denote these regions as *Core-Periphery* ($a \geq b \geq d$), *Dual-Core* ($a \geq d \geq b$), and *Random* ($b \geq a \geq d$).

 **Core-Periphery** ($a \geq b \geq d$)

In networks with this configuration, at the high-level, the network can be divided into two groups. One group is dense with many connections as value a is the largest; fewer connections exist within the nodes in the other group as highlighted by value d; and moderate connections exist between nodes from different groups. Many real-world networks exhibit a core-periphery structure [47], where they form a core group and another group which acts as its periphery. Value a represents the *core strength*.

 **Dual-Core** ($a \geq d \geq b$)

In this configuration, each group is internally well-connected but the connections between the two groups are sparse. We denote this configuration as the *Dual-Core* structure. Basically, the two groups of nodes form two major cores of the network, of which one exhibits a stronger core strength, and they are relatively independent of each other. Values a and d represent the core strength of each group.

 **Random** ($b \geq a \geq d$)

This configuration is quite different from the previous two. Essentially, one can not find two recursive groups with more connections within each group than across groups. To some extent, it is indication that there is not much difference in the importance, or "coreness" among nodes. This reminds us of random graphs, such as those generated by the Erdös-Rényi $G(n, p)$ model [48], where a random network of n nodes is created in which every edge exists with an equal probability p. To validate our speculation, we generate many random networks by fixing the number of nodes $n = 1024$ and varying the probability p. We compute the Kronecker points of these networks to obtain a, b, and d values. Figure 2.6 illustrates that for a random network, we almost always have $b \geq a \geq d$, unless

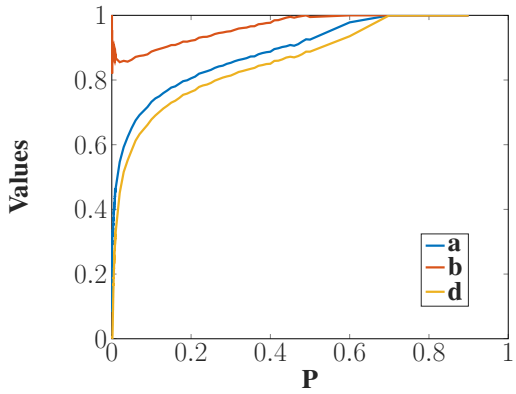


Figure 2.6: Distribution of Kronecker points (a, b, d) for Random Networks $G(n, p)$. Here, $n = 1024$. We observe that in random graphs, unless the graph is really dense, $b \geq a \geq d$. the graph is really dense, e.g, $p > 0.75$. We observe the same pattern for $n = 2048, 4096$ and 8192. One may note that when p is close to 0, the random network is empty, but value b does not converge to 0. This is an artifact caused by an overestimation in the KRONFIT algorithm, due to the limitations of KRONFIT we have discussed in Section 2.3.2.

For any graph or its subgraphs, their Kronecker points should be located within one of these three regions. This observation inspires us to represent a network using the location of the Kronecker points of the network and its subgraphs, e.g., a network exhibiting a core-periphery structure at the whole-network level, but most of its subgraphs are random.

2.7.2 Interpretability of Kronecker Hulls

As detailed in Section 2.7.1, a Kronecker point, representing any graph, is guaranteed to fall within one of three regions: Core-Periphery, Dual-Core, and Random, where each region represents a specific network structure. This property allows one to describe the whole network, its subgraph(s), or a 3D space within its Kronecker hull.

We demonstrate the interpretability of Kronecker Hulls by analyzing our networks. For each network, Table 2.3 provides the regions in which the whole graph and its 180 subgraphs are located. We make observations at the (I) whole-network or (II) subgraph levels:

I. Characterizing Networks. We identify region that the Kronecker point of the whole

Table 2.3: Subgraphs of Networks. Subgraphs can exhibit a core-periphery structure, be dual-core, or random. Here, symbol $\rightarrow$ indicates that the network structure observed (e.g., core-periphery) is the same as that of the smaller sampling proportion (to the left). A parenthesis is used to list all network structures observed at a sampling proportion.

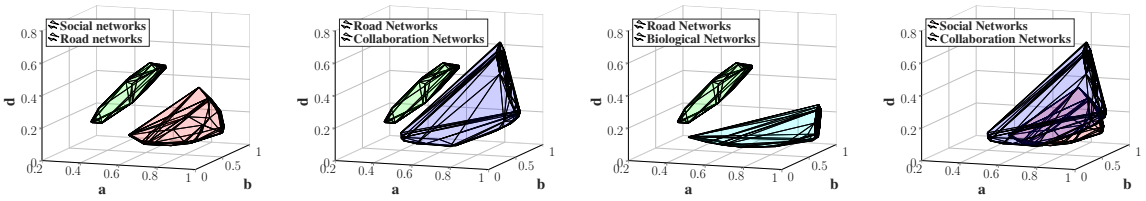
Types	Network \ Sampling Proportion	10%	20%	30%	40%	50%	60%	70%	80%	90%	Whole Graph
Social Networks	Brightkite										
	Flixster										
	Gowalla										
	Hyves										
	Livejournal										
	MySpace										
	Orkut										
	YouTube										
Collaboration Networks	Astro-Ph										
	Cond-Mat										
	Gr-Qc										
	Hep-Th										
Road Networks	Road-CA										
	Road-PA										
	Road-TX										
Biological Networks	Bio-Dmela										
	Bio-Grid-Yeast										
	Human-Brain										

graph is within. We find that (1) all social networks are in the Core-Periphery region, confirming past research indicating that social networks exhibit a core-periphery structure [47]; (2) three collaboration networks are in the Core-Periphery region, and the other is in the Dual-Core region; (3) all road networks are within the Dual-Core region, which can be explained by the fact that road networks often exhibit a recursive structure. For example, the connections between two states are sparse, relying on a few highways or trunk roads, while the connections within a state are denser. This road structure also applies to two cities within a state; (4) all biological networks are in the Core-Periphery region, confirming past research that has observed a core-periphery structure within protein-protein interaction networks [49] and human brain [50]; and (5) none is in the Random region.

II. Characterizing Subgraphs. By identifying the regions for subgraphs, we find that: (1)

Figure 2.7: Kronecker hulls across Categories

(a) Road vs. Social (b) Road vs. Collab (c) Road vs. Biological (d) Social vs. Collab



for social and biological networks, most subgraphs are in the same region in which the whole network is in: the Core-Periphery region. This observation indicates that small samples (e.g., 20%) of most social and biological networks exhibit properties similar to that of the whole network. This observation also explains our previous observation on the rapid drop of distances between sphere centers and the Kronecker point of the whole network when the sampling proportion changes from 10% to 30%. We also observe that when the sample is too small, the network core is not yet formed in some samples, leading to those samples being in the Random region; (2) for road networks, we find sampled subgraphs that are less than 50% of the network are often in the Random region, and after that exhibit a Dual-Core structure. This transition explains why the distances between sphere centers and the Kronecker point of the whole network drop sharply when the proportion is less than 60%; (3) for collaboration networks, the composition of subgraphs is complex. For large samples, subgraphs exhibit either a Core-Periphery or a Dual-Core structure. For small samples, we also observe some Random subgraphs. Also, subgraph structure strongly depends on sampled nodes. This complexity explains why Kronecker points do not cluster well in collaboration networks as same-size samples can exhibit various network structures, e.g., for being from various academic communities.

2.8 Applications

We present some applications of network shapes. In particular, we use Kronecker hulls to (A) identify the category a network belongs to (e.g., road), and (B) study the similarity between two networks.

Table 2.4: Overlap between Kronecker Hulls of Categories

	Social Networks	Collaboration Networks	Biological Networks
Social Networks	100%	75.63%	8.92%
Collaboration Networks	75.63%	100%	4.22%
Biological Networks	8.92%	4.22%	100%

2.8.1 Network Categorization

Kronecker hulls can help categorize networks, i.e., determine whether a network is a social network or a biological one. We demonstrate the feasibility of network categorization using Kronecker hulls. To categorize networks, we create a Kronecker hull for a family of graphs (e.g., all social networks). Here, for each network category (biological/social/road-/collaboration), we create a Kronecker hull from the Kronecker points (i.e., subgraphs) of all the networks within that category. As depicted in Figure 2.7, Kronecker hull of road networks is well-separated from those of the other three categories. Basically, given a Kronecker hull of one road network, or Kronecker points of some subgraphs from a road network, one can easily verify that it is not from the other three categories. For the other three categories, we compute the overlap between their corresponding Kronecker hulls. From Table 2.4, we find that biological networks have a small overlap with the other two types of networks, meaning that it is not very difficult to distinguish a biological network from a social or a collaboration network. However, the overlap between collaboration networks and social networks is large, being over 75%. We plot both Kronecker hulls in Figure 2.7d. This large overlap is not surprising, as both categories involve human social behavior. Clearly, a comprehensive supervised learning framework (e.g., that uses Kronecker hull attributes as features) can further advance network categorization.

2.8.2 Computing Network Similarity

Kronecker hulls can capture various forms of (dis) similarity between two networks:

I. Consider two large graphs A and B to be 100% similar when A is a subgraph of B. By construction, Kronecker hull of A will be within Kronecker hull of B, i.e., a 100%

Table 2.5: Kronecker Hull Overlaps for Social Networks

	Brightkite	Flixster	Gowalla	Hyves	Livejournal	MySpace	Orkut	YouTube
Brightkite	100%	0.07%	12.2%	0	0	0	0	0
Flixster	0.07%	100%	50.14%	0	0	9.25%	0	49.03%
Gowalla	12.2%	50.14%	100%	0	10.56%	0	0	11.64%
Hyves	0	0	0	100%	0	0	0	0
Livejournal	0	0	10.56%	0	100%	0	0	0
MySpace	0	9.25%	0	0	0	100%	0	16.52%
Orkut	0	0	0	0	0	0	100%	0
YouTube	0	49.03%	11.64%	0	0	16.52%	0	100%

similarity leads to 100% overlap between the corresponding Kronecker hulls. Hence, the overlap may indicate some level of similarity.

II. Consider two graphs to be similar, when they both belong to similar categories of networks (e.g., a social network is similar to a collaboration network) and dissimilar, otherwise. Our discussion in Section 2.8.1 showed that when networks belong to dissimilar categories, there is little to no overlap between their Kronecker hulls. For instance, a road network in our dataset will have no overlap with a random network from any other category, while a social network is expected to have overlap with a collaboration network.

III. Consider two networks to be similar, when they are semantically similar, e.g., both are video sharing networks. Here, we assume semantic similarity leads to some level of network structure similarity. We show that Kronecker hulls can capture some level of semantic (dis) similarity by taking social networks as an example. Table 2.5 lists the overlap between the Kronecker hulls of each pair of the eight social networks. We make the following observations: (1) various similar networks exhibit overlap. For example, Brightkite and Gowalla, both location-based social networks, overlap. Also, MySpace, YouTube and Flixster are well connected to each other, which may be explained by the content they share. MySpace has a strong music emphasis, and YouTube and Flixster are often used to share videos or music; and (2) social networks popular in specific countries (e.g., Orkut and Hyves) are well separated from other networks.

We believe these observations motivate a systematic study on the connection between graph similarity and overlap of Kronecker hulls, as part of our future work.

2.9 Additional Related Work

In addition to related research discussed throughout the chapter, our work has links to the following areas:

I. Network Visualization. Network visualization [51] aims to visualize large-scale networks in real-time to facilitate easy network exploration or specific applications, e.g., detecting users with expertise [52]. Network shapes provide a compact and interpretable way to visualize a network and its subgraphs.

II. Graph Compression. There has been an increasing interest in graph compression [53–55], especially in large-scale networks. Storing the network shape provides an alternative compact solution to graph compression. In our experiments on graphs with millions of nodes, Kronecker hulls can often be represented with less than 40 boundary points.

2.10 Conclusions

We propose network shapes and a linear algorithm to construct one type of network shapes: Kronecker Hulls. A Kronecker hull represents a network as a convex hull. Kronecker hulls are compact, easy to visualize, and capture various properties of a network and its subgraphs. We demonstrate the interpretability of Kronecker points, as a Kronecker point, representing any graph, is guaranteed to fall within one of three regions: Core-Periphery, Dual-Core, and Random. With this property, Kronecker hulls can be used to characterize graphs (e.g., how does a 10% subgraph look like?), Moreover, we find that network shapes can be used for applications such as network categorization (e.g., is this a social or a biological network?), and for computing graph similarity.

Chapter 3

Network-based Biometrics

3.1 Introduction

In Chapter 2, we demonstrate that network shapes can be used to characterize graphs, to categorize networks, and to compute network similarity. In this chapter, we show more applications of network shapes and we extend biometrics studies to network data, by proposing two new problems in network studies: network identification and network authentication. As we know, networks research has often been conducted on anonymized graphs, especially for social networks, as data privacy is critical. To protect the users' privacy while preserving network properties, anonymization techniques have been widely used before publishing social network data [56, 57]. To validate the authenticity of such anonymized graphs, it is natural to ask questions such as: Given a large graph G, can we verify that it is a Facebook graph but not collected from Twitter, or a biological network? Can we identify the source of an anonymized network, i.e., its *network identity*? To answer these questions, the first natural solution that comes to mind is to check whether a network contains a subgraph that is isomorphic to G. The problem is called *subgraph isomorphism*, and is known to be NP-complete [58], so solving it is infeasible for most large networks. Hence, we need an alternative solution that is reasonably accurate and highly efficient.

Problem Formulation. To identify a person, two types of systems have been designed in biometrics literature: (1) *identification* systems and (2) *authentication* systems [59]. An identification system recognizes a subject without the subject claiming an identity, i.e., "Who am I?". It tries to match the subject with everyone enrolled in the system database and obtains the best match. On the other hand, an authentication system either

rejects or accepts the submitted claim of identity, i.e., "Am I who I claim to be?". In spite of their differences, sometimes the terms authentication and identification are used interchangeably [59]. Inspired by biometrics research, we formulate two new problems:

1. **Network Identification**. Given a set of networks $N = \{N_1, N_2, ..., N_n\}$, and a subgraph G sampled from some $N_i \in N$ using sampling strategy S, we want to identify G, i.e., the network N_i from which G is sampled.

2. **Network Authentication** (or *network identity-authentication*). Subgraph G is claimed to be sampled from a certain network N_i via sampling strategy S. The authentication system either accepts or rejects this claim.

Following the problem formulation, our first aim is to build an identity to represent a network, similar to how a fingerprint represents a person. We propose two ways to build a network identity:

1. **Embedding-based Identity**. Intuitively, one can represent a network using a feature vector or its graph embedding. Graph embedding methods aim to map a graph into a low-dimensional vector that preserves the network structure [8]. Hence, one can represent the identity of a network N_i with its embedding, and match the embedding of subgraph G with that of other networks.

2. **Distribution-based Identity**. One limitation of the embedding-based identity is that it is not unique, as graph embedding methods generally do not guarantee uniqueness for different networks. Hence, inspired by *ridge-based representation* [60] for fingerprints, we propose *distribution-based identity*. The ridge-based representation is one of the most widely-used representations for fingerprints and it is based on the hypothesis that ridge structures (*minutiae*, e.g. ridge ending and ridge bifurcation) and their distributions are distinct across fingerprints. It inspires us to, instead of using one embedding, represent a network identity as the distribution of embedding values

for subgraphs of a network, so that the identity is unique and can preserve subgraph information. These settings are a perfect match with network shapes.

The Present Work. We introduce network identification and network authentication with the following contributions:

1. **Network Identity**. We introduce the *network identity* and two identity types: *embedding-based identity* and *distribution-based identity*. We prove that the embedding-based identity can capture graph structure information and/or other relationships between samples (subgraphs) and the source network. We demonstrate that the distribution-based identities are unique by showing that for real-world networks the similarity among such identities for various networks is generally low. Our distribution-based identities are visualizable in 3D and are easy to interpret; hence, we show examples on how the structural differences in networks are reflected in their identities. We evaluate the two types of identities in both identification/authentication problems.

2. **Network Identification**. We introduce two methods to predict the network from which a graph is sampled using the developed network identities. The first is a supervised learning method, which is highly accurate (84.4%). We also introduce an easier to implement method that relies on the distances between the sample embedding to the network identities, achieving a 70.8% accuracy.

3. **Network Authentication**. We propose two techniques to solve the network authentication problem: a *supervised splitter*, which has a low equal error rate, and a *Voronoi splitter*, which allows controlling the false reject with an acceptable false accept rate across networks.

The rest of the chapter is organized as follows. In Section 3.2, we introduce two types of network identities, and connect the identities with the relationships between the sample subgraphs and source network. The data used in our experiments is summarized in Section

3.3. We discuss the uniqueness of identities and partial identity in Section 3.4. We propose methods to solve network identification in Section 3.5, and network authentication in Section 3.6. An application in biometrics is explored in Section 3.7 and the limitations of our work is discussed in Section 3.8. We review the related work in Section 3.9 and conclude in Section 3.10.

3.2 Network Identity

The first step to identify a graph is to build an identity for it. We propose two types of network identities: (1) embedding-based identity and (2) distribution-based identity.

3.2.1 Embedding-based Identity

In theory, any embedding method that can preserve network structural information and the similarity between samples (subgraphs) and the network from which they are sampled from can be used as an embedding-based identity. Here, we choose *Kronecker points* as the embedding method and prove its utility for both network authentication and identification. We focus on the connection between Kronecker Points and graph similarity.

Kronecker Points and Graph Similarity. Graph kernels have been traditionally used to measure the similarity between two graphs [61]. Here, we prove that if Kronecker points of two graphs are more similar (i.e., closer in the 3D space), the graphs are expected to have higher similarity in terms of the random-walk graph kernel between them [62].

Theorem 3.2.1 (Kronecker Initiator and Graph Kernel)**.** *For graphs G_1 and G_2 generated by probability matrices $\mathcal{P}_1$ and $\mathcal{P}_2$, where $\mathcal{P}_1$ and $\mathcal{P}_2$ are the k^{th} Kronecker power of Kronecker initiator matrices Θ_1 and Θ_2, i.e., $\mathcal{P}_1 = \Theta_1^{\otimes k}, \mathcal{P}_2 = \Theta_2^{\otimes k}$, the expected random-walk graph kernel between G_1 and G_2 is lower bounded by the product of the ℓ_1-norm of*

Θ_1 and Θ_2, and the product of the sizes of G_1 and G_2:

$$\mathbb{E}(\mathcal{K}(G_1, G_2)) \geq (|G_1||G_2|+|\lambda|(\|\Theta_1\|_1\|\Theta_2\|_1)^k)^{-1}, \tag{3.1}$$

where λ denotes the decay factor of the graph kernel.

Proof. Random-walk graph kernel performs random walks on both graphs and counts the number of matching walks, discounting longer walks. Random-walk graph kernel [62] is formulated as $\mathcal{K}(G_1, G_2) = \frac{1}{|G_1||G_2|}\boldsymbol{e}^T(I - \lambda A_\times)^{-1}\boldsymbol{e}$, where $\boldsymbol{e}$ denotes the all 1 vector, λ denotes the decay factor of the graph kernel, I is the identity matrix, and $A_\times$ is the adjacency matrix of the direct product graph of G_1 and G_2, i.e., $A_\times = A_{G_1} \otimes A_{G_2}$. Hence, $I, A_\times \in \mathbb{R}^{|G_1||G_2|\times|G_1||G_2|}$,

$$\begin{aligned}
\mathcal{K}(G_1, G_2) &= \frac{1}{|G_1||G_2|}\boldsymbol{e}^T(I - \lambda A_\times)^{-1}\boldsymbol{e} \\
&= \frac{1}{|G_1||G_2|}\|(I - \lambda A_\times)^{-1}\|_1 \\
&\geq \frac{1}{|G_1||G_2|}\frac{\|I\|_1}{\|I - \lambda A_\times\|_1} \\
&= \frac{1}{\|I - \lambda A_\times\|_1} \\
&\geq \frac{1}{\|I\|_1+|\lambda|\|A_\times\|_1}.
\end{aligned}$$

As $\|A_\times\|_1 = \|A_{G_1} \otimes A_{G_2}\|_1 = \|A_{G_1}\|_1\|A_{G_2}\|_1$ and $\mathbb{E}(\|A_{G_1}\|_1) = \|\Theta_1\|_1^k$ and $\mathbb{E}(\|A_{G_2}\|_1) = \|\Theta_2\|_1^k$, using Jensen's inequality, we get

$$\begin{aligned}
\mathbb{E}(\mathcal{K}(G_1, G_2)) &\geq \mathbb{E}(\frac{1}{\|I\|_1+|\lambda|\|A_\times\|_1}) \\
&\geq (|G_1||G_2|+|\lambda|(\|\Theta_1\|_1\|\Theta_2\|_1)^k)^{-1}.
\end{aligned}$$

□

Corollary 3.2.1.1 (Kronecker Points and Graph Kernel)**.** *In Theorem 3.2.1, when Θ_1 and*

Θ_2 *are* 2×2 *Kronecker initiator matrices,* $\mathbb{E}(\mathcal{K}(G_1, G_2)) \geq (|G_1||G_2|+|\lambda|(\langle\Theta_1, \Theta_2\rangle_F + \frac{3}{2}(\|\Theta_1\|_2^2+\|\Theta_2\|_2^2))^k)^{-1}$*, where* $\langle\cdot,\cdot\rangle_F$ *denotes Frobenius inner product.*

Proof. Assume $\Theta_1 = \begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix}$ and $\Theta_2 = \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix}$. $\|\Theta_1\|_1\|\Theta_2\|_1 = (a_1 + b_1 + c_1 + d_1)(a_2 + b_2 + c_2 + d_2) = \sum_{x\in\{a,b,c,d\}} x_1x_2 + \sum_{\substack{x,y\in\{a,b,c,d\} \\ x\neq y}} x_1y_2 \leq \langle\Theta_1, \Theta_2\rangle_F + \sum_{\substack{x,y\in\{a,b,c,d\} \\ x\neq y}} \frac{x_1^2+y_2^2}{2} = \langle\Theta_1, \Theta_2\rangle_F + \frac{3}{2}(\|\Theta_1\|_2^2+\|\Theta_2\|_2^2)$. □

Corollary 3.2.1.1 indicates that two graphs are expected to be more similar if their Kronecker initiator matrices have larger inner products.

Kronecker Points of Sample Subgraphs. Next, we demonstrate that Kronecker points can preserve the relationships between sampled subgraphs and the network from which they are sampled. In Theorem 3.2.2, we prove that if a network and its subgraph are perfectly fitted by two Kronecker initiator matrices, then the Euclidean distance between their Kronecker points is well bounded, corroborating previous empirical findings that Kronecker points of large sampled subgraphs are close to that of the whole network [12]. Theorem 3.2.3 gives the error bound of the fitting process using KRONFIT algorithm.

Theorem 3.2.2 (Kronecker Points of Samples)**.** *For network* $G = (V, E), |V| = 2^k$ *generated by a Stochastic Kronecker graphs probability matrix* $\mathcal{P} = \Theta^{\otimes k}$ *and its subgraph* G_s *sampled using Random Node Sampling with sampling proportion* p *where* $p > 0.5$*,*[1] *the expected* ℓ_1*-norm of the difference of their adjacency matrices is* $\mathbb{E}(\|A_G - A_{G_s}\|_1) = (1 - p^2)\|\Theta\|_1^k$.

Corollary 3.2.2.1. *If a network* G *and its subgraph* G_s *are perfectly fitted by two Kronecker initiator matrices* Θ *and* Θ_s*,* $\|\Theta - \Theta_s\|_1 = \sqrt[k]{1 - p^2}\|\Theta\|_1$.

Corollary 3.2.2.1 bounds the distance between Kronecker points of a graph and its subgraph, assuming perfect fit. It also indicates that the Kronecker points of small subgraphs

[1] The condition $p > 0.5$ ensures that the size of the subgraph is greater than 2^{k-1}, and when performing the fitting, KRONFIT will add isolated nodes so that the number of nodes becomes 2^k [41].

can be far away from the source network. However, a real-world graph can rarely be perfectly fit by Kronecker initiators. In Theorem 3.2.3 we will discuss the bounds on fitting error.

Theorem 3.2.3 (Error Bound on Fitting Real-World Graphs)**.** *For graph $G = (V, E)$ with $|V| = 2^k$, fitting the most likely Kronecker initiator matrix Θ provides an upper bound on the expected error $\mathbb{E}(\|A_G - \mathcal{P}\|_1)$.*

Proof. Denote σ as a node mapping from A_G to $\mathcal{P}$ and $(A_G - \mathcal{P})_\sigma$ the difference between A_G and $\mathcal{P}$, given σ. Then,

$$\begin{aligned}
\|(A_G - \mathcal{P})_\sigma\|_1 &= \sum_{(u,v)\in E} (1 - \mathcal{P}[\sigma_u, \sigma_v]) + \sum_{(u,v)\notin E} \mathcal{P}[\sigma_u, \sigma_v] \\
&= |V|^2 - (\sum_{(u,v)\in E} \mathcal{P}[\sigma_u, \sigma_v] + \sum_{(u,v)\notin E} (1 - \mathcal{P}[\sigma_u, \sigma_v])) \\
&\leq |V|^2 - |V|^2 (\prod_{(u,v)\in E} \mathcal{P}[\sigma_u, \sigma_v] \prod_{(u,v)\notin E} (1 - \mathcal{P}[\sigma_u, \sigma_v]))^{\frac{1}{|V|^2}} \\
&= |V|^2 (1 - (\prod_{(u,v)\in E} \mathcal{P}[\sigma_u, \sigma_v] \prod_{(u,v)\notin E} (1 - \mathcal{P}[\sigma_u, \sigma_v]))^{\frac{1}{|V|^2}}).
\end{aligned}$$

As $\prod_{(u,v)\in E} \mathcal{P}[\sigma_u, \sigma_v] \prod_{(u,v)\notin E} (1 - \mathcal{P}[\sigma_u, \sigma_v])$ is the likelihood $P(G|\mathcal{P}, \sigma)$ in Stochastic Kronecker Graphs [38], so

$$\begin{aligned}
\mathbb{E}(\|A_G - \mathcal{P}\|_1) &= \sum_\sigma \|(A_G - \mathcal{P})_\sigma\|_1 P(\sigma) \\
&\leq |V|^2 (1 - \sum_\sigma P(G|\mathcal{P}, \sigma)^{\frac{1}{|V|^2}} P(\sigma)) \\
&= |V|^2 (1 - \mathbb{E}(P(G|\mathcal{P})^{\frac{1}{|V|^2}})).
\end{aligned}$$

KRONFIT estimates the initiator matrix by maximizing $\mathbb{E}(P(G|\mathcal{P}))$; hence, KRONFIT provides an approximation on the upper bound of $\mathbb{E}(\|A_G - \mathcal{P}\|_1)$. □

Alternative Estimation of Kronecker Points. In some cases, the KRONFIT algorithm may lead to over/underestimation. When the number of nodes within a real-world network is not a power of 2, KRONFIT will add isolated nodes so that the number of nodes becomes a power of 2 [41]. Adding isolated nodes may lead to underestimation of the parameters as it decreases the overall edge density and core strength of the groups. On the other hand, as the input to KRONFIT is a list of edges, when the network is extremely sparse and the graph size is small, KRONFIT can overestimate, as it overlooks real isolated nodes within the network [12]. Therefore, in this chapter, we use another estimator of Kronecker initiator matrix for comparison. Instead of maximizing the likelihood, the method-of-moments estimator [63] minimizes the difference between the counts for edges, triangles, wedges, and 3-stars of a real graph and the expected counts of the fitted Kronecker graph. Compared to KRONFIT, it gets closer to the counts of these local structures. However, our experiments show that in general the Kronecker points estimated by the method-of-moments estimator lead to less classification performance on graphs. In the rest of the chapter, by default, we consider Kronecker points estimated by KRONFIT, and we will explicitly mention, if the method-of-moments estimator is used.

3.2.2 Distribution-based Identity

Here, we aim to represent a network identity with the distribution of embedding values for subgraphs of a network. Therefore, we construct the distribution-based identity based on the network representation we proposed in Chapter 2: **Network Shapes**. Here, we build a network shape, more specifically the **Kronecker Hull** for each network and use it as its distribution-based identity, with the same setup in 2.5. The Kronecker hull is used as the distribution-based identity.

3.3 Data Description

For our experiments, we use the same network dataset in Section 2.5, and we add two more networks as follows:

- *Road-BEL* [43]: Belgium's OpenStreetMap road network.
- *Human-Brain* [64]: the network of human brain.

In total, we have twenty real-world networks from four general network categories: social networks, collaboration networks, road networks, and biological networks.

3.4 Uniqueness and Partial Network Identity

3.4.1 Uniqueness of Network Identity

Uniqueness is a basic requirement for an identity. As mentioned, graph embedding methods do not generally guarantee uniqueness, which is also true for Kronecker points. Hence, we check whether distribution-based identity can capture the distinctiveness of networks. We define the distribution-based identity similarity and investigate the similarity between identities of different networks.

Distribution-based Identity Similarity. To view how similar two distribution-based identities are, let us take a look at an example first. We define the similarity between

Table 3.1: Distribution-based Identity Similarity

Types	Network	Brightkite	Flixster	Gowalla	Hyves	Livejournal	MySpace	Orkut	YouTube	Astro-Ph	Cond-Mat	Gr-Qc	Hep-Th	Road-BEL	Road-CA	Road-PA	Road-TX	Bio-Dmela	Bio-Grid-Human	Bio-Grid-Yeast	Human-Brain
Social Networks	Brightkite	1	0	0.04	0	0	0	0	0	0	0	0	0	0	0	0	0	0.09	0.18	0	0
	Flixster	0	1	0.12	0	0	0.05	0	0.22	0.07	0	0	0.01	0	0	0	0	0	0	0	0
	Gowalla	0.04	0.12	1	0	0.01	0.01	0	0.04	0.12	0.01	0.01	0.02	0	0	0	0	0	0.01	0	0
	Hyves	0	0	0	1	0	0.01	0	0	0	0.03	0.02	0.04	0	0	0	0	0	0	0	0
	Livejournal	0	0	0.01	0	1	0	0	0	0.04	0	0	0	0	0	0	0	0	0	0	0
	MySpace	0	0.05	0.01	0.01	0	1	0	0.07	0.04	0.03	0.02	0.03	0	0	0	0	0	0	0	0
	Orkut	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	YouTube	0	0.22	0.04	0	0	0.07	0	1	0.05	0.01	0.01	0.02	0	0	0	0	0	0	0	0
Collaboration Networks	Astro-Ph	0	0.07	0.12	0	0.04	0.04	0	0.05	1	0.08	0.05	0.05	0	0	0	0	0	0	0	0
	Cond-Mat	0	0	0.01	0.03	0	0.03	0	0.01	0.08	1	0.45	0.57	0	0	0	0	0	0	0	0
	Gr-Qc	0	0	0.01	0.02	0	0.02	0	0.01	0.05	0.45	1	0.43	0	0	0	0	0	0	0	0
	Hep-Th	0	0.01	0.02	0.04	0	0.03	0	0.02	0.05	0.57	0.43	1	0	0	0	0	0	0	0	0
Road Networks	Road-BEL	0	0	0	0	0	0	0	0	0	0	0	0	1	0.23	0	0.13	0	0	0	0
	Road-CA	0	0	0	0	0	0	0	0	0	0	0	0	0.23	1	0.22	0.48	0	0	0	0
	Road-PA	0	0	0	0	0	0	0	0	0	0	0	0	0	0.22	1	0.25	0	0	0	0
	Road-TX	0	0	0	0	0	0	0	0	0	0	0	0	0.13	0.48	0.25	1	0	0	0	0
Biological Networks	Bio-Dmela	0.09	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0.49	0	0
	Bio-Grid-Human	0.18	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0.49	1	0	0
	Bio-Grid-Yeast	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0.02
	Human-Brain	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.02	1

identities using Jaccard Index:

$$\mathsf{similarity}(A, B) = \frac{\mathsf{volume}(\mathsf{ID}_A \cap \mathsf{ID}_B)}{\mathsf{volume}(\mathsf{ID}_A \cup \mathsf{ID}_B)}\,, \tag{3.2}$$

where volume is the volume of a distribution-based identity, and ID_A and ID_B represent identities of networks A and B, respectively. It is easy to find that $\mathsf{volume}(\mathsf{ID}_A \cup \mathsf{ID}_B) = \mathsf{volume}(\mathsf{ID}_A) + \mathsf{volume}(\mathsf{ID}_B) - \mathsf{volume}(\mathsf{ID}_A \cap \mathsf{ID}_B)$, and $\mathsf{volume}(\mathsf{ID}_A \cap \mathsf{ID}_B)$ is easy to calculate as intersection of convex sets is convex. Table 3.1 lists the similarity between all pairs of the identities (the results have been rounded to two digits). We observe that (1) similarity between most identities (i.e., shapes) is small, i.e., below 0.1; (2) networks from different categories in general have very low similarity. Road networks and biological networks are not similar to networks from other categories, while social networks and collaboration networks have some similarity; (3) within the same category, some similarity exists. For example, the similarity between YouTube and Flixster is 0.22, road networks are relatively similar to each other, and three collaboration networks are also relatively similar. In general, the highest similarity is 0.57, which does not violate the uniqueness of the network identity across different networks. Note that we do not claim absolute uniqueness for the distribution-based identity, as it is built using graph sampling; however, we assume that the distribution of embedding values for subgraphs can capture the distinctness of the network identities. Moreover, previous studies [12] have shown that points representing samples of the same proportion exhibit a clustering phenomenon, indicating the stability of a distribution-based identity to some extent, which we also observe in our experiments (see Sections 3.5 and 3.6). When we estimate Kronecker points using the method-of-moments estimator, we find similar patterns but the similarity between the network identities are generally higher (see Table 3.2).

Table 3.2: Distribution-based Identity Similarity (using method-of-moments estimator)

Types	Network	Brightkite	Flixster	Gowalla	Hyves	Livejournal	MySpace	Orkut	YouTube	Astro-Ph	Cond-Mat	Gr-Qc	Hep-Th	Road-BEL	Road-CA	Road-PA	Road-TX	Bio-Dmela	Bio-Grid-Human	Bio-Grid-Yeast	Human-Brain
Social Networks	Brightkite	1	0.06	0.03	0.02	0	0.03	0.13	0	0	0.02	0.03	0	0	0	0	0	0.06	0.28	0	0
	Flixster	0.06	1	0	0.07	0	0	0	0	0	0	0	0	0	0	0	0	0.01	0.04	0	0
	Gowalla	0.03	0	1	0	0.01	0.21	0.06	0	0	0	0	0	0	0	0	0	0	0	0	0
	Hyves	0.02	0.07	0.01	1	0	0.03	0	0	0	0	0	0	0	0	0	0	0.01	0.03	0	0
	Livejournal	0	0	0.01	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	MySpace	0.03	0	0.21	0.03	0	1	0.03	0	0	0	0	0	0	0	0	0	0	0	0	0
	Orkut	0.13	0	0.06	0	0	0.03	1	0	0	0	0	0	0	0	0	0	0	0.02	0	0
	YouTube	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Collaboration Networks	Astro-Ph	0	0	0	0	0	0	0	0	1	0.02	0.03	0	0	0	0	0	0.06	0.02	0.01	0
	Cond-Mat	0.02	0	0	0	0	0	0	0	0.02	1	0.48	0.05	0	0	0	0	0.49	0.15	0	0
	Gr-Qc	0.03	0	0	0	0	0	0	0	0.03	0.48	1	0.12	0	0	0	0	0.49	0.13	0	0
	Hep-Th	0	0	0	0	0	0	0	0	0	0.05	0.12	1	0.16	0.20	0.03	0.13	0.03	0	0	0
Road Networks	Road-BEL	0	0	0	0	0	0	0	0	0	0	0	0.16	1	0.77	0.08	0.78	0	0	0	0
	Road-CA	0	0	0	0	0	0	0	0	0	0	0	0.20	0.77	1	0.15	0.75	0	0	0	0
	Road-PA	0	0	0	0	0	0	0	0	0	0	0	0.03	0.08	0.15	1	0.13	0	0	0	0
	Road-TX	0	0	0	0	0	0	0	0	0	0	0	0.13	0.78	0.75	0.13	1	0	0	0	0
Biological Networks	Bio-Dmela	0.06	0.01	0	0.01	0	0	0	0	0.06	0.49	0.49	0.03	0	0	0	0	1	0.25	0	0
	Bio-Grid-Human	0.28	0.04	0	0.03	0	0	0.02	0	0.02	0.15	0.13	0	0	0	0	0	0.25	1	0	0
	Bio-Grid-Yeast	0	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0	1	0.09
	Human-Brain	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.09	1

3.4.2 Partial Distribution-based Network Identity

Theoretically, one can sample all the possible subgraphs from a network to build a distribution-based identity that is "complete" (similar to how one can have a high resolution fingerprint scan). However, this is inefficient. Let us assume the network identity we have constructed is practically "complete". A few questions comes up: How sensitive is a network identity to the number of sample points taken? Due to the definition of convex hull, if we take a subset of Kronecker points of the complete network identity to build a partial network identity, the partial network identity should also be a subset of the complete network identity. In other words, the complete network identity shrinks to a partial network identity. How different are the complete network identity and a partial one? To answer these questions, we study the partial network identity by varying the sampling step size s and the number of independent samples for each proportion t, and check the similarity between the partial identities and the complete one. We first fix $s = 10\%$ and vary t from 5 to 20. Figure 3.1 indicates that the distribution-based identity is in general insensitive to t as (1) for the smallest $t = 5$, the similarity is over 50%; (2) for most networks, by sampling 13 to 14 subgraphs for each proportion, we can create a partial network identity that is 90% similar to the complete network identity. This means taking fewer samples of the

Figure 3.1: Distribution-based Identity Change with t

(a) Social Networks (b) Collab. Networks (c) Road Networks (d) Bio. Networks

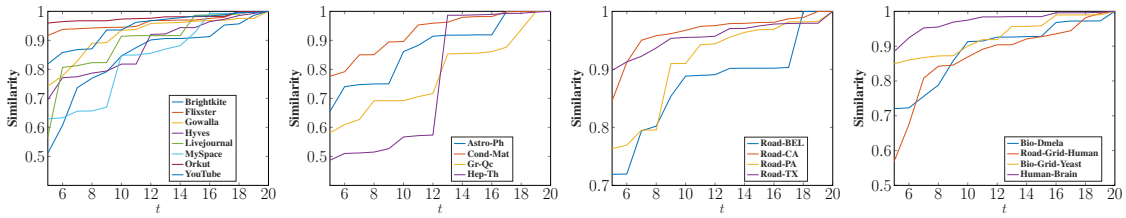
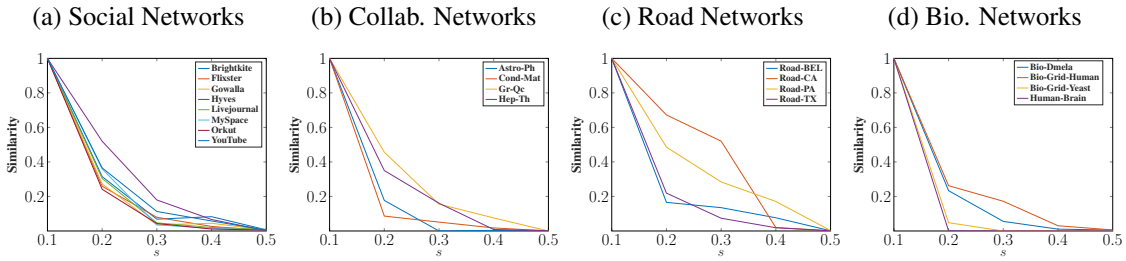


Figure 3.2: Distribution-based Identity Change with s



same proportion will not change the identity much; the observation follows that of [12], where Kronecker points representing samples of the same proportion exhibit a clustering phenomenon. Next, we fix $t = 20$, and vary the step size s from 10% to 50%. Figure 3.2 shows that the network identity is more sensitive to s as (1) the similarity drops quickly with the increase of sampling step size, and (2) the similarity drops to 0 as the volume turns to 0 (when the identity becomes 2-dimensional). The observation shows that by setting s smaller and taking more samples of different sizes will lead to a more stable network identity. For the partial identities using the method-of-moments estimator, the patterns are similar, but in general the partial identities are more sensitive to the change of t and s. In Sections 3.5 and 3.6, we will discuss the performance of the partial identity for the network identification/authentication problems.

3.5 Network Identification

3.5.1 Experimental Setup

From each network, we sample many subgraphs representing graphs G which are to be identified/authenticated. We vary the sampling proportion from 10% to 99% and sample using random node sampling. For each proportion, we sample two subgraphs. Hence, for each network we have $90 \times 2 = 180$ subgraphs, and for 20 networks, we have $180 \times 20 = 3,600$ samples to be identified/authenticated.

3.5.2 Identification with Embedding-based Identity

To use the embedding-based identity for identification, we embed both G and all other N_i as Kronecker points. We consider the identification problem in the following way: Given the n (=20) identities of N_i's, we *split* the whole embedding space, the $1 \times 1 \times 1$ cube, into n regions, so that each region represents the embedding space for the samples of a certain network. In our work, we propose two splitters.

Voronoi Splitter. It calculates the Euclidean distance between the Kronecker point of a graph G to that of all other networks (N_i's) and reports the closest N_i as the identified network. This is equivalent to building a *Voronoi diagram* [65] for the set of Kronecker points of all N_i's, where the *Voronoi cell* for N_j denotes the set of graphs identified as N_j.

Supervised Splitter. Instead of reporting the closest N_i, for each sample G, we use the 20 distances (from a sample to each N_i) as features, and the name of the networks as the class label, to train a multiclass classification model. In this experiment, we use 10-fold cross validation, and decision tree, linear SVM, k-NN and bagged trees as our classifiers.

We provide four baselines for comparison.

1. **Top Eigenvalues**. Top eigenvalues have been used to study graph similarity [66]. We compute the top 5 eigenvalues of each sample as features for classification. The time complexity of the method is $\mathcal{O}(n^2)$, where n is the number of nodes.

2. **Truncated Spectral Moments**. The spectral moments of the random walk transition matrix of a network have been proven to be closely related to the network structure and various network properties [15]. Therefore, we compute the truncated (first four) spectral moments of each sample as features for classification. We use the APPROX-SPECTRALMOMENT algorithm proposed by [67] to compute the accurate estimates of the low-order moments. The algorithm estimates the moments by simulating many random walks and computes the proportion of closed walks. To compute the ℓ-th spectral

Table 3.3: Network Identification Accuracy with Embedding-based Identity

Type	Voronoi Splitter	Supervised Splitter	Baselines			
			Top Eigenvalues	Truncated Spectral Moments	Graph2Vec	Random Prediction $(1/n)$
All Networks	40.2%	**84.0%**	62.4%	82.0%	81.7%	5%
Social Networks	50.3%	94.2%	74.8%	95.5%	83.7%	12.5%
Collaboration Networks	58.2%	78.8%	70.9%	94.8%	97.4%	25%
Road Networks	32.9%	67.0%	44.9%	51.2%	86.3%	25%
Biological Networks	66.5%	98.3%	90.4%	99.7%	89.9%	25%

moment by simulating s random walks, it takes $O(s\ell)$ time.

3. **Graph2Vec**. GRAPH2VEC is a graph embedding technique, which views a graph as a document and the rooted subgraphs around each node as words. It extends document embedding neural networks to embed a graph as a vector [68].

4. **Random Prediction**. A simple *random prediction*, so the accuracy will be $1/n$, where n is the number of networks.

We evaluate the methods for all networks and within each network category and report the results in Table 3.3. For supervised splitter, we report the result of the best classifier, as the prediction turns out to be insensitive to the choice of learning algorithm. Table 3.3 illustrates that (1) both Voronoi splitter and Supervised Splitter outperform the random prediction; (2) Voronoi splitter does not perform as well as the other baselines. This is not surprising, as Theorem 3.2.2 has shown that when the sampling proportion p is small, the Kronecker point of the sample can be far from that of the whole network; (3) Supervised Splitter performs best and achieves an overall accuracy of 84.0%. It is slightly better than Truncated Spectral Moments and GRAPH2VEC, and significantly better than the other methods; and (4) the performance on road networks is not as good as other categories, while GRAPH2VEC performs relatively stable across different categories. Comparing both methods, we find that (1) Voronoi Splitter is simple and does not need a training process, but can make more mistakes, especially on smaller samples; (2) Supervised Splitter performs better as it learns from the distances from the samples to different networks, making more informed decisions. Table 3.4 lists the result of using the method-of-moments esti-

Table 3.4: Network Identification Accuracy with Embedding-based Identity (method-of-moments estimator)

Type	Voronoi Splitter	Supervised Splitter	Baselines			
			Top Eigenvalues	Truncated Spectral Moments	Graph2Vec	Random Prediction $(1/n)$
All Networks	37.7%	61.6%	62.4%	82.0%	81.7%	5%
Social Networks	53.5%	67.3%	74.8%	95.5%	83.7%	12.5%
Collaboration Networks	39.9%	75.3%	70.9%	94.8%	97.4%	25%
Road Networks	24.0%	35.1%	44.9%	51.2%	86.3%	25%
Biological Networks	69.4%	88.1%	90.4%	99.7%	89.9%	25%

mator. The result is not as good as using KRONFIT, but is still comparable with the top eigenvalues baseline.

3.5.3 Identification with Distribution-based Identity

To use the distribution-based identity for identification, we follow a roadmap similar to that of the embedding-based method. The difference is that the network identity N_i is represented as a 3D shape. Therefore, we need to define the distance between a 3D point and a 3D shape. Considering definitions of the distance between two sets of points and geometrical properties of a convex polyhedron, we consider the following three Euclidean distances as candidates:

1. $\mathsf{d}_{\mathsf{shortest}}$. $\mathsf{d}_{\mathsf{shortest}}$ is defined based on the shortest distance between two points from sets A and B, respectively:

$$d(A, B) = \inf\{d(x, y) | x \in A, y \in B\}. \tag{3.3}$$

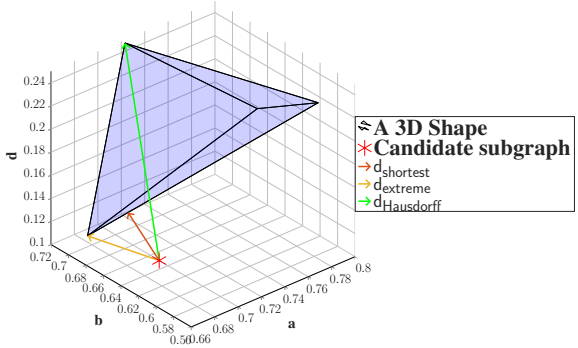


Figure 3.3: Three Distances between a 3D point and a 3D Shape.

Table 3.5: 90th Percentile of the Distance Distribution

Type	Network	$\mathsf{d}_{\mathsf{shortest}}$	$\mathsf{d}_{\mathsf{extreme}}$	$\mathsf{d}_{\mathsf{Hausdorff}}$
Social Networks	Brightkite	0.0139	0.0861	0.6017
	Flixster	0.0149	0.0344	0.3476
	Gowalla	0.0257	0.0494	0.5898
	Hyves	0.0149	0.0331	0.5218
	Livejournal	0.0069	0.0247	0.1547
	MySpace	0.0130	0.0325	0.4399
	Orkut	0.0055	0.0161	0.1713
	YouTube	0.0138	0.0394	0.4687
Collaboration Networks	Astro-Ph	0.0196	0.0572	0.5938
	Cond-Mat	0.0189	0.0670	1.0400
	Gr-Qc	0.0143	0.0818	1.1514
	Hep-Th	0.0147	0.0394	1.0730
Road Networks	Road-BEL	0.0306	0.0598	0.8940
	Road-CA	0.0182	0.0595	0.7866
	Road-PA	0.0297	0.0917	0.8327
	Road-TX	0.0220	0.0812	0.7651
Biological Networks	Bio-Dmela	0.0087	0.0413	0.7175
	Bio-Grid-Human	0.0127	0.0489	0.5932
	Bio-Grid-Yeast	0.0040	0.0483	0.2489
	Human-Brain	0.0063	0.0248	0.1341

In our case, it refers to the distance from a point to the closest point on the surface (all the facets) of the shape if the point is outside the shape; otherwise it is 0.

2. $\mathsf{d}_{\mathsf{Hausdorff}}$. Hausdorff distance is used to measure how far two sets A and B are in a metric space:

$$d_H(A, B) = \max\{\sup_{a\in A}\inf_{b\in B} d(a, b), \sup_{b\in B}\inf_{a\in A} d(a, b)\}. \tag{3.4}$$

It is the largest of the distances from a point in one set to the closest point in the other and is commonly used in computer vision research [69]. In our case, $\mathsf{d}_{\mathsf{Hausdorff}}$ refers to the distance from a point to the farthest boundary point (i.e., extreme points) of the shape.

3. $\mathsf{d}_{\mathsf{extreme}}$. All boundary points of a network shape are some of the Kronecker points of samples used for generating the shape. Hence, we also use the distance from a point to the closest boundary point of the shape.

Figure 3.3 is a simple example to illustrate these three distances. For each network and the test samples drawn from it, we list the 90th percentile of the distances distribution in Tables 3.5 and 3.6. Based on the definitions, we know that $\mathsf{d}_{\mathsf{shortest}} \leq \mathsf{d}_{\mathsf{extreme}} \leq \mathsf{d}_{\mathsf{Hausdorff}}$. From

Table 3.6: 90th Percentile of the Distance Distribution (method-of-moments estimator)

Type	Network	$\mathsf{d}_{\mathsf{shortest}}$	$\mathsf{d}_{\mathsf{extreme}}$	$\mathsf{d}_{\mathsf{Hausdorff}}$
Social Networks	Brightkite	0.0143	0.0480	0.3896
	Flixster	0.0104	0.0308	0.5672
	Gowalla	0	0.0510	0.2606
	Hyves	0	0.0307	0.5497
	Livejournal	0	0.0459	0.0957
	MySpace	0	0.0324	0.1788
	Orkut	0.0171	0.0372	0.1825
	YouTube	0	0.0272	0.1253
Collaboration Networks	Astro-Ph	0.0176	0.0501	0.2997
	Cond-Mat	0.0357	0.0695	0.8505
	Gr-Qc	0.0263	0.0790	0.9066
	Hep-Th	0.0375	0.1074	1.1391
Road Networks	Road-BEL	0.0071	0.3314	1.6682
	Road-CA	0.0295	0.5531	1.6576
	Road-PA	0.0299	0.1365	1.6636
	Road-TX	0.0502	0.5129	1.6470
Biological Networks	Bio-Dmela	0.0208	0.0678	0.7309
	Bio-Grid-Human	0.0215	0.0760	0.5702
	Bio-Grid-Yeast	0.0050	0.0806	0.4375
	Human-Brain	0.0115	0.0321	0.1902

Table 3.7: Network Identification Accuracy with Distribution-based Identity

Type	Voronoi Splitter				Supervised	Baselines			
	$\mathsf{d}_{\mathsf{shortest}}$	$\mathsf{d}_{\mathsf{extreme}}$	$\mathsf{d}_{\mathsf{hausdorff}}$	$\mathsf{d}_{\mathsf{weighted}}$	Splitter	Top Eigenvalues	Truncated Spectral Moments	Graph2Vec	Random Prediction $(1/n)$
All Networks	61.6%	63.8%	16.8%	70.8%	**84.4%**	62.4%	82.0%	81.7%	5%
Social Networks	81.3%	81.4%	25.7%	86.7%	96.4%	74.8%	95.5%	83.7%	12.5%
Collaboration Networks	65.7%	63.7%	25%	75.0%	84.2%	70.9%	94.8%	97.4%	25%
Road Networks	48.8%	46.7%	35%	52.4%	76.8%	44.9%	51.2%	86.3%	25%
Biological Networks	70.8%	63.1%	34.6%	76.3%	80.4%	90.4%	99.7%	89.9%	25%

the table, we observe that most of the Kronecker points of the subgraphs are around the surface and the boundary of the network shape of the source network. For most networks, $\mathsf{d}_{\mathsf{Hausdorff}}$ is large, especially for collaboration networks, which indicates that different subgraphs of the same network can be far from each other.

Next, we use the three distances with the two splitters we used in the last section for identification. To make our approach clear, we provide an example in Figure 3.4. We report the result in Table 3.7. We find that for Voronoi Splitter, compared with the embedding-based identity, the distribution-based identity with $\mathsf{d}_{\mathsf{shortest}}$ and $\mathsf{d}_{\mathsf{extreme}}$ improves performance significantly. It can outperform both Top Eigenvalues and Random Prediction, as the distribution-based identity can preserve subgraph information. It is not surprising that $\mathsf{d}_{\mathsf{Hausdorff}}$ does not perform well as it can be explained by our observation and discussion of the 90th percentile of the distance distribution. Based on these observations, we consider

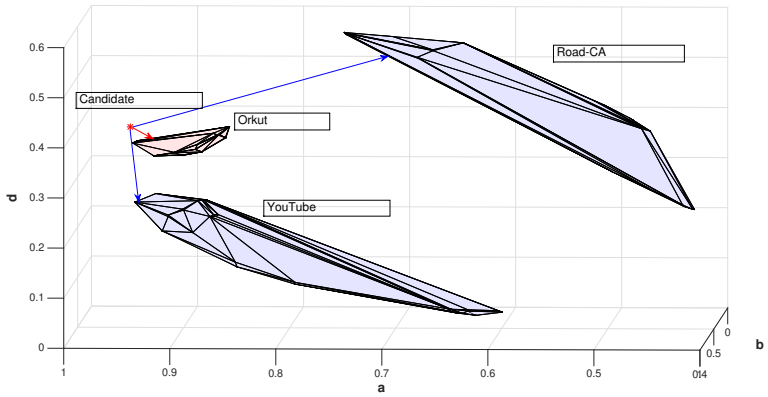


Figure 3.4: **An example of the two splitters**. Here, we have three distribution-based identities in our database: Orkut, YouTube, and road network of California. Our goal is to identify a candidate graph, which is the red point. To identify, we check the distances between the red point and the three 3D shapes. For Voronoi splitter, we pick the closest shape, in this case Orkut, as its identity. For supervised splitter, we use distances to all network identities as features and the network name as the label to train a classifier.

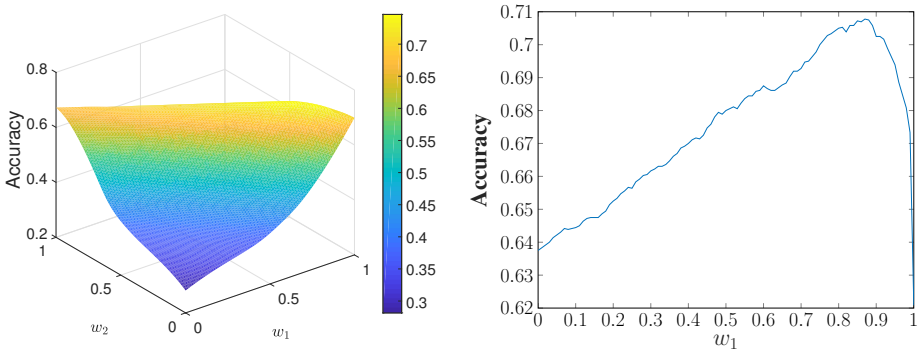
using a combination of these three distances for identification. We use the weighted average $\mathsf{d}_{\mathsf{weighted}} = w_1 \times \mathsf{d}_{\mathsf{shortest}} + w_2 \times \mathsf{d}_{\mathsf{extreme}} + w_3 \times \mathsf{d}_{\mathsf{Hausdorff}}$, where $w_1 + w_2 + w_3 = 1$. To get the best weights, one can use supervised learning for learning the weights. Here, for simplicity, we do grid search on the feasible weights w_1, w_2, w_3 and plot the accuracy change in Figure 3.5a. The plot shows that the accuracy is high when $w_1 + w_2 \approx 1$ and it drops as w_3 increases. The best accuracy is 70.8% with $w_1 = 0.87, w_2 = 0.13, w_3 = 0$. Figure 3.5b provides the accuracy change when w_3 is set to 0, i.e., $w_1 + w_2 = 1$. We find the accuracy increases quickly when w_1 increases from 0 to 0.7 and drops quickly when w_1 is greater than 0.9. Based on the observations, we set $\mathsf{d}_{\mathsf{weighted}} = 0.87 \times \mathsf{d}_{\mathsf{shortest}} + 0.13 \times \mathsf{d}_{\mathsf{extreme}}$, and in general $\mathsf{d}_{\mathsf{weighted}}$ performs best among these distances.

For Supervised Splitter, we use $\mathsf{d}_{\mathsf{shortest}}$, $\mathsf{d}_{\mathsf{extreme}}$, and $\mathsf{d}_{\mathsf{Hausdorff}}$ as features. Each graph G has $3 \times 20 = 60$ features for all networks, and we use the name of the networks as the class labels. Table 3.7 shows that compared with the Embedding-based identity, the performance slightly improves and it reaches an overall accuracy 84.4%.

We conduct the same experiments by using the method-of-moments estimator, and Table 3.8 indicates that it performs slightly better than the Embedding-based identity, but not as

Table 3.8: Network Identification Accuracy with Distribution-based Identity (method-of-moments estimator)

Type	Voronoi Splitter				Supervised Splitter	Baselines			
	$\mathsf{d}_{\text{shortest}}$	$\mathsf{d}_{\text{extreme}}$	$\mathsf{d}_{\text{hausdorff}}$	$\mathsf{d}_{\text{weighted}}$		Top Eigenvalues	Truncated Spectral Moments	Graph2Vec	Random Prediction $(1/n)$
All Networks	45.9%	39.3%	20.2%	51.5%	64.1%	62.4%	82.0%	81.7%	5%
Social Networks	52.5%	41.5%	28.1%	53.7%	68.5%	74.8%	95.5%	83.7%	12.5%
Collaboration Networks	51.4%	52.5%	31.9%	67.8%	80.1%	70.9%	94.8%	97.4%	25%
Road Networks	36.7%	26.8%	23.3%	32.6%	43.9%	44.9%	51.2%	86.3%	25%
Biological Networks	77.5%	76.7%	25%	88.6%	89.6%	90.4%	99.7%	89.9%	25%

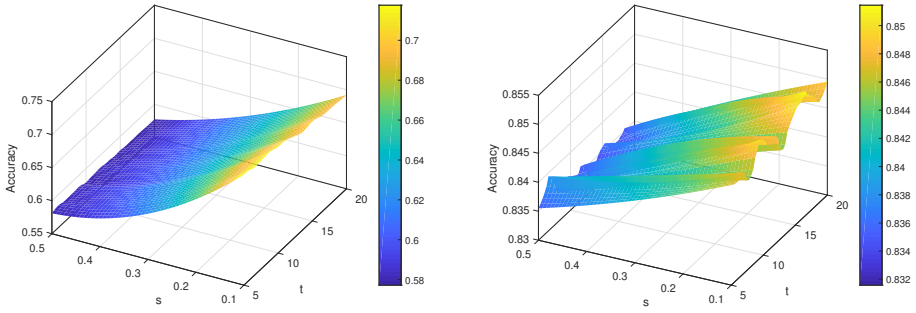


(a) $w_1 + w_2 + w_3 = 1$ (b) $w_1 + w_2 = 1, w_3 = 0$

Figure 3.5: Accuracy with Weighted Distance $\mathsf{d}_{\text{weighted}}$

good as using KRONFIT.

Identification with Partial Network Identity. As discussed in Section 3.4, partial distribution-based network identity can be constructed similar to the complete network identity by taking fewer sample subgraphs. We investigate how effective partial distribution-based identities are in the network identification task. Based on the previous study on the similarity of partial network identity and complete network identity, we speculate that the network identification accuracy is more sensitive to the change of the sampling step size s. Figure 3.6a and 3.6b illustrate the accuracy change of Voronoi splitter (using $\mathsf{d}_{\text{weighted}}$) and Supervised Splitter respectively with different s and t configurations. In general, the accuracy does not change with the number of samples t for each proportion and it slightly drops with the increase in sampling step size s.



(a) Voronoi Splitter (b) Supervised Splitter

Figure 3.6: **Prediction Performance with Partial Identity**. (1) The Supervised Splitter is robust to the change of both t and s. The accuracy does not change with the number of samples t for each proportion and it slightly drops with the increase in sampling step size s from 85% to 83%. (2) Similar patterns are observed for Voronoi Splitter. Differently, the accuracy decreases more with the increase of s, from 70% to 58%.

3.6 Network Authentication

For network authentication, given the distance from the identity of G to that of a network N_i, we aim to accept or reject the claim that G is sampled from N_i.

3.6.1 Authentication

Different from network identification, for network authentication, we need to split the whole embedding space into two regions: the *accept* and *reject* regions. We also propose two methods: a Voronoi splitter and a supervised splitter.

Voronoi Splitter. For the embedding-based identity, we use the r-percentile of the distances from the Kronecker points of samples to that of the source network as a threshold. If the distance between identities of G and N_i is less than threshold d, we accept the claim; otherwise, we reject it. An advantage of this method is that we can control the false reject rate (FRR) of the authentication system, e.g., in one experiment, we set r = 90, so FRR is fixed at 10%. It allows one to have a geometric interpretation of this splitter. That is, we create a ball centered at the Kronecker point of the network with a diameter equal to $2 \times d$. Everything inside the ball (the boundary included) will be accepted and everything outside

Table 3.9: Authentication with Voronoi Splitter ($r = 90$)

Type	Networks	Embedding-based			Distribution-based ($\mathsf{d}_{\mathsf{shortest}}$)		
		Accuracy	AUC	FAR	Accuracy	AUC	FAR
Social Networks	Brightkite	39.83%	0.61	62.51%	97.81%	0.94	1.73%
	Flixster	61.67%	0.67	38.92%	90.77%	0.90	9.18%
	Gowalla	48.72%	0.67	53.27%	78.52%	0.84	22.16%
	Hyves	39.50%	0.54	62.16%	99.04%	0.95	0.42%
	Livejournal	84.58%	0.62	11.81%	98.52%	0.95	0.98%
	MySpace	55.72%	0.70	45.82%	94.32%	0.92	5.42%
	Orkut	85.31%	0.45	10.20%	99.44%	0.95	0.00%
	YouTube	48.08%	0.64	53.71%	91.57%	0.91	8.33%
Collaboration Networks	Astro-Ph	40.08%	0.59	40.08%	77.50%	0.83	23.24%
	Cond-Mat	35.00%	0.56	67.37%	78.77%	0.84	21.90%
	Gr-Qc	5.08%	0.50	99.91%	79.58%	0.84	21.04%
	Hep-Th	21.17%	0.47	81.67%	84.54%	0.87	15.78%
Road Networks	Road-BEL	5.03%	0.48	99.80%	88.33%	0.89	11.75%
	Road-CA	16.67%	0.41	86.02%	90.70%	0.90	9.28%
	Road-PA	7.42%	0.43	96.49%	89.20%	0.90	10.85%
	Road-TX	8.06%	0.47	95.79%	89.97%	0.90	10.03%
Biological Networks	Bio-Dmela	37.97%	0.46	62.89%	97.96%	0.94	1.57%
	Bio-Grid-Human	36.58%	0.54	65.32%	92.80%	0.91	7.05%
	Bio-Grid-Yeast	88.31%	0.84	11.17%	98.18%	0.94	1.34%
	Human-Brain	95.19%	0.93	4.53%	98.95%	0.95	0.52%

Table 3.10: Authentication with Voronoi Splitter (moment based $r = 90$)

Type	Networks	Embedding-based			Distribution-based ($\mathsf{d}_{\mathsf{shortest}}$)		
		Accuracy	AUC	FAR	Accuracy	AUC	FAR
Social Networks	Brightkite	87.00%	0.46	8.42%	89.81%	0.90	10.20%
	Flixster	89.14%	0.47	6.17%	98.39%	0.94	1.17%
	Gowalla	90.22%	0.47	5.03%	82.44%	0.88	18.19%
	Hyves	93.81%	0.49	1.26%	92.78%	0.93	7.19%
	Livejournal	91.75%	0.48	3.42%	95.56%	0.96	4.47%
	MySpace	91.53%	0.48	3.65%	84.80%	0.90	15.82%
	Orkut	88.11%	0.48	7.46%	94.11%	0.92	5.67%
	YouTube	94.78%	0.50	0.23%	90.86%	0.93	9.33%
Collaboration Networks	Astro-Ph	88.56%	0.48	6.96%	89.86%	0.90	10.15%
	Cond-Mat	25.97%	0.60	77.78%	84.97%	0.87	15.29%
	Gr-Qc	26.44%	0.55	76.70%	84.03%	0.87	16.29%
	Hep-Th	22.75%	0.59	81.23%	94.47%	0.92	5.29%
Road Networks	Road-BEL	9.28%	0.32	93.30%	89.14%	0.90	10.91%
	Road-CA	5.03%	0.50	99.94%	86.58%	0.88	13.60%
	Road-PA	7.44%	0.37	95.79%	86.81%	0.88	13.36%
	Road-TX	5.00%	0.50	100.00%	86.81%	0.88	13.36%
Biological Networks	Bio-Dmela	31.28%	0.52	70.99%	86.47%	0.88	13.71%
	Bio-Grid-Human	41.22%	0.61	60.99%	84.19%	0.87	16.11%
	Bio-Grid-Yeast	43.67%	0.70	59.30%	93.47%	0.92	6.35%
	Human-Brain	97.81%	0.94	1.78%	99.14%	0.95	0.38%

the ball is rejected. For the distribution-based identity, we know from the distribution of $\mathsf{d}_{\mathsf{shortest}}$, $\mathsf{d}_{\mathsf{extreme}}$, and $\mathsf{d}_{\mathsf{Hausdorff}}$ for samples of each network that most points are around the surface of the network shape; hence, we can use the r-percentile of the distances to the surfaces as the threshold. Similarly, one can interpret the splitter as creating a band around the surface of the distribution-based identity with a diameter equal to $2 \times d$, accepting everything inside the band and rejecting everything outside. Table 3.9 shows that the method does not work well with embedding-based identity, but performs well with distribution-based identity. The false accept rate (FAR) varies from 0% to more than 20%, and for most networks it is below 10%. When we use method-of-moments estimator, the result does not change much (see Table 3.10). Moreover, we vary r, when we use the distribution-based identity, and plot the change of average FAR and FRR across networks in Figure 3.7, and it turns out that $r = 90$ leads to the equal error rate.

Supervised Splitter. For distribution-based identity, we use $\mathsf{d}_{\mathsf{shortest}}$, $\mathsf{d}_{\mathsf{extreme}}$, and $\mathsf{d}_{\mathsf{Hausdorff}}$

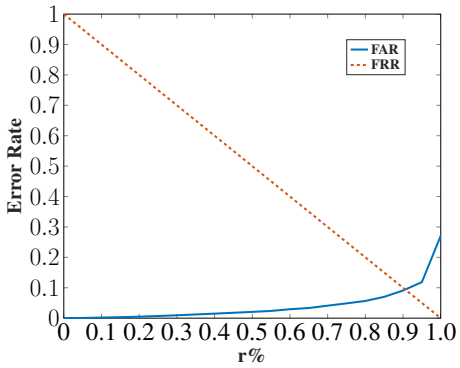


Figure 3.7: **Equal Error Rate**. (1) With the increase of r (decrease of FRR), FAR increases slowly. (2) When r is 100, the FRR is around 25%. It shows that $\mathsf{d}_{\mathsf{surface}}$ is a good indicator as the majority of the samples from other networks are far away. (3) when r is 90, the FAR and FRR are equal, which leads to the equal error rate.

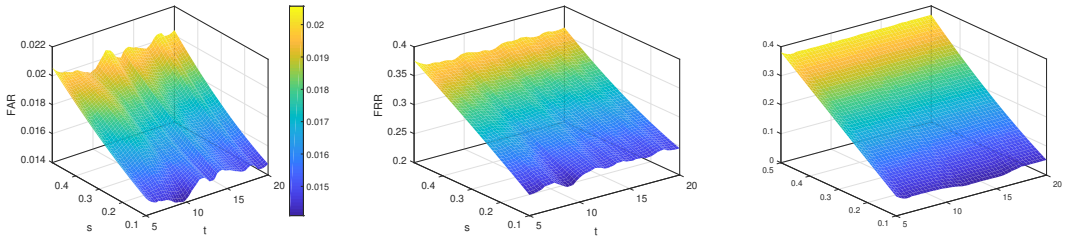
Table 3.11: Authentication with Supervised Splitter

Classifier	Embedding-based	Distribution-based
Decision Tree	0.21 (0.02)	0.07 (0.01)
k-NN	0.21 (0.02)	0.09 (0.01)
SVM	0.28 (0.02)	0.10 (0.01)

Note: Mean and standard deviation of the EERs across networks.

between identities of G and N_i as three features, and whether G is sampled from N_i as a binary label. We train a supervised learning classifier with 10-fold cross validation for each network. For the embedding-based identity, we use the distance between the Kronecker points of G and N_i as the only feature. We report Equal Error Rate (EER), at which the false accept rate (FAR) is equal to the false reject rate (FRR), in Table 3.11. The results show that classifiers using both identities have a low EER indicating a reasonable performance, especially for using distribution-based identity. Comparing the two splitters, one can see a trade-off between the FAR and the FRR.

Authentication with Partial Network Identity. Next, we use partial distribution-based network identity for network authentication. Figures 3.8a and 3.8b illustrate the change of the average FAR and FRR with the partial network identities for Supervised Splitter. We notice that both FAR and FRR slowly increase with the sampling step size s and do not change much with t. For Voronoi Splitter, Figure 3.8c shows the FAR quickly increases with the increase in sampling step size s. This can be explained by the fact that with the increase of s, the partial network identity rapidly shrinks as its similarity to the complete



(a) FAR of Supervised Splitter

(b) FRR of Supervised Splitter

(c) FAR of Voronoi Splitter

Figure 3.8: **Authentication Performance with Partial Identity**. (a-b) For Supervised Splitter, both FAR and FRR increase with the sampling step size s slowly and they do not change much with t. FAR is generally lower than 2% and FRR increases from 20% to over 35%; (c) For Voronoi Splitter, we use the distances to the surfaces and set the percentile $r = 90$, so FRR is fixed at 10%. Similarly, we find the splitter is not sensitive to t, but the FAR quickly increases with the increase in sampling step size s.

network identity drops fast, which in turn leads to the 90th percentile d becoming large. A large threshold d will accept more false samples. In this case, we need to find the equal error rate for the partial network identity to strike a balance between FAR and FRR.

3.7 Application to Biometrics

As we have shown, our methods can be used for network identification and network authentication. Here, we explore whether they can help with real-world biometrics. Here, we use two touch-based biometrics datasets to authenticate user identity: (1) *La. Tech Touch Dataset.* This dataset was collected by Abdul Serwadda and his colleagues [70]. The dataset contains touch data for 138 users on a phone screen. The subjects were students, faculty, or staff at Louisiana Tech University; (2) *SU Touch Dataset.* The second dataset is collected by us. It includes the touch data on both a phone screen and a tablet screen from 116 students at Syracuse University. For both datasets, users were asked to answer multiple questions through Android applications. Users had to scroll/swipe back and forth to find the answers to the questions, and for each stroke representing the path taken by a user's finger during scrolling, the applications recorded points touched by the user's finger (the x and y coordinates) every 15 milliseconds. We aim to authenticate a user's identity

Table 3.12: User Authentication Performance

Classifier	Network Authentication	Stroke Authentication	
		Horizontal	Vertical
Decision Tree	0.30 (0.04)	0.31 (0.29)	0.38 (0.28)
k**-NN**	0.39 (0.11)	0.17 (0.14)	0.27 (0.15)
SVM	0.28 (0.04)	0.16 (0.14)	0.18 (0.14)

Mean and standard deviation of the EERs across the population. Stroke authentication separates horizontal and vertical stroke.

Table 3.13: User Authentication Performance

Classifier	Phone	Tablet
Decision Tree	0.41 (0.01)	0.43 (0.05)
k**-NN**	0.40 (0.01)	0.42 (0.04)
SVM	0.42 (0.01)	0.45 (0.04)

Mean and standard deviation of the EERs across the population.

based on her overall swipe/scroll patterns on the screen. Therefore, we represent the whole phone/tablet screen as a grid network. We set the whole screen as a grid network with each node covering a square area. For the La. Tech dataset, where the layout of the phone is 480×800, we set the node as a 4×4 square area. For each user, we create an undirected simple graph, a *swipe graph*. For each two consecutive recorded points along all the paths (strokes) of the user, we connect the two nodes (areas) where the two points are located. For each user swipe graph, we construct the distribution-based network identity and sample 180 subgraphs for testing, and we use the supervised splitter for authentication. Serwadda et al. [70] propose a stroke-based authentication by using 28 features of each stroke, including velocity and acceleration along the stroke, to authenticate user identity. Table 3.12 lists the EERs of the two methods, and the network authentication method has comparable mean EERs and smaller standard deviation indicating that the method is stable across users. For the SU dataset, two types of phones are used: Samsung Galaxy S6 whose layout is 1440×2560, where we set the node as a 12×12 square area; HTC-One whose layout is 1080×1920, where we set the node size 9×9. We set these node sizes to ensure the generated grid networks having similar size as those of the networks in the Latech dataset. Also, the layout of the tablet is 1536×2048 and we set the node as a 13×13 square area. Table 3.13 shows the EERs of the SU dataset are around 0.4 for both phone and tablet.

3.8 Limitations

We have demonstrated the effectiveness of using network identities for network identification and network authentication. However, based on the problem settings, there are a few assumptions and limitations for the methods: (1) the networks are not isomorphic, i.e., if N_i and N_j are isomorphic, then $i = j$. If two networks are isomorphic, they are basically the same graph after anonymization, and there is no way to distinguish them; and (2) subgraph G is not too small to lose its identity. Consider a small subgraph such as a triad $\triangle$, which can be found in most networks, so it does not make much sense to verify its identity; (3) while networks can be sampled using different sampling strategies, here we assume subgraphs are sampled by random node sampling, and we create distribution-based network identities using the same sampling strategy; (4) we assume that we have the whole networks, so we can build ground-truth network identities in the system; and (5) we do not consider fraud in the identity collection process.

3.9 Additional Related Work

Additionally, our work has links to the following areas:

I. Subgraph Isomorphism. Subgraph isomorphism problem has been long studied in graph theory. As solving subgraph isomorphism leads to the maximum clique problem and testing the existence of a Hamiltonian cycle in a graph, it is an NP-complete problem [58]. Under certain conditions (e.g. the subgraph is a planar graph), the complexity of the problem can be reduced to linear time [71]. However, the conditions are hard to fulfill for large networks, which is the main motivation for us to investigate identification/authentication methods. As we mentioned before, the time complexity to compute the network identity (both embedding-based and distribution-based) is linear in the number of nodes n and edges m of the network.

II. Identification and Authentication. In biometrics, many human physiological or behavioral characteristic are used to facilitate identification, such as facial information [72], iris [73], or gait [74]. Theoretically, if one converts these characteristic to network data, our graph-based method can be applied leading to graph-based biometrics.

IV. Network Categorization and Network Classification. Network categorization aims to predict the category (or domain) of a network, and most network classification studies have been focused on the classification of graphs within a particular category such as molecular graphs [75, 76]. C. James et al. [77] use 12 graph features, such as density, number of triangle, and the like to predict the category of a network with a high accuracy. Different from them, we are trying to identify whether a graph is sampled from a network not a general category. However, these methods may help us improve our identification systems introduced here by conducting network categorization as a preprocessing step.

3.10 Conclusions

To the best of our knowledge, we are the first to extend biometrics studies to network data by formulating the network identification and network authentication problems. In this study, we propose and compare two types of network identities, and we demonstrate their utility in solving both problems. The embedding-based identity is easy to construct, but the distribution-based identity performs better with simple methods. For network identification, we propose two techniques to predict the network from which a graph is sampled. The supervised learning method is highly accurate, and a simple method that uses only one Euclidean distance has a reasonable accuracy. For network authentication, we show that the supervised method yields a low equal error rate, and the Voronoi method enables controlling the false reject rate, while attaining a reasonable false accept rate across networks. We show that our graph-based methods can also be used for biometrics, authenticating users based on their touch data on devices.

Chapter 4

Embedding Networks with Spectral Moments

4.1 Introduction

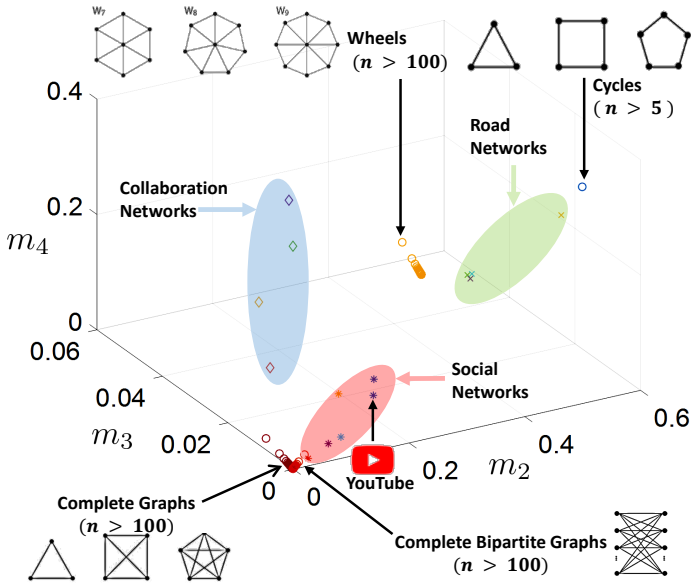


Figure 4.1: The Zoo of Networks

In the previous chapters, we use Kronecker Points as the embedding method to build a network shape: a Kronecker hull. A Kronecker hull is interpretable as it reveals the relationship between the network and its subgraphs and Kronecker points can capture three global network structures: Core-Periphery, Dual-Core and Random (or no core). However, the interpretability of Kronecker points is limited as (1) it can hardly show network properties other than its core strength; (2) it is computed by fitting the network model (Stochastic Kronecker Graphs) to a network, which may not be accurate for real networks. Therefore, we aim to look for a 3D embedding method with more interpretability, in other

words, we want the embedding method captures as much graph information as possible with only three values. In this way, we provide a interpretable, visualizable, and compact *embedding space* for networks. We denote such embedding space as "a zoo of networks", where one can easily characterize networks based on their locations in this zoo, similar to how animals are grouped and located in different regions of a zoo. Figure 4.1 illustrates a big picture of this zoo of networks obtained using the method developed in this chapter. In the figure, we plot the embeddings of real-world networks from three different categories: Social Networks, Collaboration Networks and Road Networks, and also plot various types of graphs including complete graphs, cycles, complete bipartite graphs, and wheels of different sizes (the number of nodes n are mostly above 100).

To build such an embedding space, we need the network embedding method to meet the following criteria: (1) **Easy to visualize.** We want an embedding space that can be easily visualized, so a 3D embedding of networks is needed; (2) **Capture network structure.** The embedding values should help users understand the network structure; (3) **Capture network properties.** The embedding values should shed light on different network properties such as the degree distribution or network connectivity; and (4) **Easy and fast to compute.** The method should be scalable for large networks. Spectral graph theory can help satisfy these constraints.

Spectral graph theory connects the structure of a network to the eigenvalues and eigenvectors of its associated matrices such as the adjacency matrix or the Laplacian. The extreme eigenvalues and associated eigenvectors are often used by various spectral methods. For example, the ratio between the largest and smallest eigenvalues can help estimate the chromatic number [78, 79]; the second-smallest eigenvalue of a graph Laplacian is related to graph connectivity and the associated eigenvector is used for spectral clustering [80]. Recently, more attention is paid to the overall distribution of eigenvalues, also known as the *spectral density* of the graph. Dong et al. [28] use methods from condensed matter physics to study spectral densities in networks, and they show that the spectral density is a prac-

tical tool to analyze large real-world networks. Inspired by the strong power of spectral density analysis, we aim to find a succinct way to represent the spectral density, so as to represent the network. Naturally, we propose using spectral moments, as in statistics, moments are often used to capture the shape of a distribution. Specifically, we use the spectral moments of the random walk transition matrix as the embedding method, as (1) they have a very clear meaning, which is the expected return probability of a random walk; (2) these spectral moments, as we will see in the rest of the chapter, are closely related to the network structure and various network properties such as the degree distribution and clustering coefficient; (3) using a few of these spectral moments (*truncated moments*), more specifically, the second m_2, third m_3, and fourth m_4 moments (the first moment is 0 as we only look into undirected graphs without self-loops), we can have a 3D embedding of a network that can be visualized. We will show that the error of using truncated spectral moments is bounded. We denote this 3D embedding as the *spectral point* of the network; (4) by definition, these spectral moments are all between 0 to 1, so we can have a compact embedding space for all possible graphs, which is a $1 \times 1 \times 1$ cube (see Figure 1). The points in Figure 4.1 are spectral points. In Section 4.5, we will go into further details on how graphs can be represented using spectral points.

Overall, our contributions are mainly the following:

1. Network Embedding with Spectral Moments. We introduce *Spectral Point*, a 3D network embedding method that uses the truncated spectral moments of the network. Spectral points have the following advantages: (i) each dimension is closely related to the network structure and various network properties, so it is easy to interpret; (ii) the embedding space can help characterize various types of networks; (iii) the embedding space provides easy network visualization; and (iv) the embeddings are easy to compute.

2. Spectral Moments and Network Structure. To the best of our knowledge, we are the first to study relationship between the spectral moments of the random walk transition

matrix (more importantly and equivalently, the spectral moments of the normalized Laplacian matrix) and network structure. We connect the spectral moments to basic subgraphs such as triangles and squares.

3. Spectral Moments and Network Properties. We find that the spectral moments provide various bounds on network properties such as the degree distribution and the global clustering coefficient. We define a measure that assesses global connectivity using the spectral moments, and we prove the relationship between the spectral moments of a network and those of its connected components.

4. Representing Various Graph Types. We mathematically derive spectral moments for various types of graphs such as complete graphs, cycles, star graphs, and complete bipartite graphs. For k-regular graphs, we derive the exact value for the second moment and the range of values that the third and fourth moments can take.

5. Representing Real-World Graphs. We compute the spectral moments of real-world graphs from various categories and show that their structure and properties identified in past research is captured by spectral moments. We show that spectral moments can help get a quick understanding of a real-world network at hand.

6. Spectral Network Identification. We demonstrate that spectral moments can be used for network identification, i.e., identifying the source of an anonymized graph. Our results indicate that truncated spectral moments do not lose much predictive power.

The rest of the chapter is organized as follows. We first detail the preliminaries and notations used in the chapter in Section 4.2. In Section 4.3, we provide proofs on the relationship between spectral moments and network structure, and Section 4.4 demonstrates the relationship between spectral moments and network properties. In Section 4.5, we analyze special graphs and real-world networks using their spectral moments. In Section 4.6, we use spectral moments for network identification. After reviewing additional related work

in Section 4.7, we conclude the chapter in Section 4.8.

4.2 Preliminaries and Notation

For an undirected graph $G = (V, E)$ with vertices $V = \{v_1, v_2, ..., v_n\}$ and edges $E \subseteq V \times V$, its adjacency matrix $A \in \mathbb{R}^{n \times n}$ has $A_{ij} = 1$ if $(i, j) \in E$ and otherwise, $A_{ij} = 0$. The degree matrix $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix with node degrees on its diagonal, i.e. $D_{ii} = \sum_{j=1}^{n} A_{ij}$. Various properties of graph G (e.g., connectivity or cuts) can be identified using matrices defined in terms of A and D. The normalized Laplacian of G is the matrix $L = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$. The spectrum of a matrix is the set of its eigenvalues. The normalized Laplacian has a bounded spectrum, i.e. $0 = \mu_1 \leq \mu_2 \leq \ldots \leq \mu_{n-1} \leq \mu_n \leq 2$, where μ_i's are the eigenvalues of L. The transition matrix of the random walk on G is matrix $P = AD^{-1}$. As P is a stochastic matrix, its spectrum is also bounded: $1 = \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{n-1} \geq \lambda_n \geq -1$, where λ_i's are the eigenvalues of P. As P is *similar* to $D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ (i.e., they have the same eigenvalues), it is easy to find the relationship between the eigenvalues of P and L: $\lambda_i = 1 - \mu_i$, for $1 \leq i \leq n$.

In this work, we denote the ℓ-th spectral moment m_ℓ of a graph G using the spectrum of its random walk transition matrix P, $m_\ell = \mathbb{E}(\lambda^\ell)$, as $\frac{1}{n} \sum_{i=1}^{n} {\lambda_i}^\ell = \mathbb{E}(\lambda^\ell)$. We look at the relationship between the spectral moments and network structure and use the first few spectral moments to represent networks.

4.3 Relationship Between Spectral Moments and Network Structure

In this section, we aim to see how spectral moments are related to network structure. As we only look into undirected graphs without self-loops, it is easy to see that the first spectral

moment m_1 is 0. Therefore, we start with the second spectral moment.

4.3.1 Second Spectral Moment

Theorem 4.3.1. *The 2nd spectral moment m_2 of P is*

$$m_2 = \mathbb{E}(\lambda^2) = \mathbb{E}(1/l),$$

where l follows $p(l|k)$, the probability that a random neighbor of a node with degree k has degree l.

Proof. The ℓ-th spectral moment of P can be viewed as the expected return probability of an ℓ-step random walk starting from node i, where i is chosen uniformly at random from all the nodes [67]. For any node i, its return probability of a 2-step random walk is equal to $\sum_{j:j\sim i} \frac{1}{d_i \cdot d_j}$, where d_i and d_j are the degrees of nodes i and j, respectively, and $j \sim i$ means that j is a neighbor of i. Hence, the 2nd spectral moment of P is equal to $\mathbb{E}_{i\in V}(\sum_{j:j\sim i} \frac{1}{d_i \cdot d_j})$, and

$$\begin{aligned}
\mathbb{E}_{i\in V}(\sum_{j:j\sim i} \frac{1}{d_i \cdot d_j}) &= \mathbb{E}_{i\in V}(\frac{1}{d_i} \cdot \sum_{j:j\sim i} \frac{1}{d_j}) \\
&= \mathbb{E}_{i\in V}(\frac{1}{d_i} \cdot d_i \cdot \mathbb{E}_{j:j\sim i}(\frac{1}{d_j})) \\
&= \mathbb{E}_{i\in V}(\mathbb{E}_{j:j\sim i}(\frac{1}{d_j})).
\end{aligned}$$

Note that d_i follows the degree distribution of the graph $p(k)$, and d_j follows the conditional degree distribution $p(l|k)$ as j is constrained to be a neighbor of i, so

$$\mathbb{E}_{i\in V}(\mathbb{E}_{j:j\sim i}(\frac{1}{d_j})) = \mathbb{E}_{i\in V}(\sum_{d_j} \frac{1}{d_j} \cdot p(l = d_j|k = d_i)) \tag{4.1}$$

$$= \sum_{d_i} p(k = d_i) \cdot \sum_{d_j} \frac{1}{d_j} \cdot p(l = d_j|k = d_i) \tag{4.2}$$

$$= \sum_{d_i} \sum_{d_j} p(k = d_i) \cdot \frac{1}{d_j} \cdot p(l = d_j | k = d_i) \tag{4.3}$$

$$= \sum_{d_i} \sum_{d_j} \frac{1}{d_j} \cdot p(l = d_j, k = d_i) \tag{4.4}$$

$$= \mathbb{E}(\frac{1}{d_j}). \tag{4.5}$$

□

Note that in Theorem 4.3.1, $p(l, k)$ is the joint degree distribution of nodes (k) and their neighbors (l) and the 2nd spectral moment is $\mathbb{E}(\frac{1}{d_j})$ under this distribution. This joint distribution is not symmetric, i.e., $p(l = d_j, k = d_i) \neq p(l = d_i, k = d_j)$, and can be cumbersome to compute; hence, the following theorem states the connection between the 2nd spectral moment P and the joint degree distribution $p(d_i, d_j)$, which is symmetric.

Theorem 4.3.2. *The 2nd spectral moment m_2 of P is*

$$m_2 = \mathbb{E}(\lambda^2) = \mathbb{E}(d_i)\, \mathbb{E}(\frac{1}{d_i d_j}),$$

where $\mathbb{E}(d_i)$ denotes the average degree in the graph and $d_i d_j$ follows the joint degree distribution $p(d_i, d_j)$: the probability that a node with degree d_i is connected to another node with degree d_j.

Proof. Denote the joint degree distribution as

$$p(d_i, d_j) = \frac{n_{d_i, d_j}}{\mathbb{E}(d_i) n}, \tag{4.6}$$

where n_{d_i, d_j} is the number of edges between nodes with degree d_i and nodes with degree d_j, n is the total number of nodes in the graph, and $\mathbb{E}(d_i)$ is the average degree. Let n_{d_i}

denote the number of nodes with degree d_i. Then, $p(l = d_j, k = d_i)$ can be stated as

$$p(l = d_j, k = d_i) = p(l = d_j | k = d_i) p(k = d_i) \tag{4.7}$$
$$= \frac{n_{d_i,d_j}}{d_i n_{d_i}} p(k = d_i) \tag{4.8}$$
$$= \frac{p(d_i, d_j)\,\mathbb{E}(d_i) n}{d_i n_{d_i}} p(k = d_i) \tag{4.9}$$
$$= \frac{p(d_i, d_j)\,\mathbb{E}(d_i)}{d_i p(k = d_i)} p(k = d_i) \tag{4.10}$$
$$= \frac{p(d_i, d_j)\,\mathbb{E}(d_i)}{d_i}, \tag{4.11}$$

using which Equation (4.4) can be restated as

$$\sum_{d_i} \sum_{d_j} \frac{1}{d_j} \cdot p(l = d_j, k = d_i) = \sum_{d_i} \sum_{d_j} \frac{1}{d_j} \cdot \frac{p(d_i, d_j)\,\mathbb{E}(d_i)}{d_i} \tag{4.12}$$
$$= \mathbb{E}(d_i)\,\mathbb{E}(\frac{1}{d_i d_j}). \tag{4.13}$$

□

One can interpret Theorem 4.3.2 in this way: the expected return probability of a 2-step random walk is equal to (I) the average return probability through an edge, multiplied by (II) the average number of edges a node has, as $\frac{1}{d_i d_j}$ is the return probability of a 2-step random walk through a specific edge linking two nodes with degrees d_i and d_j, respectively, and $\mathbb{E}(\frac{1}{d_i d_j})$ is the average return probability over all edges. This observation motivates us to extend Theorem 4.3.2 to higher moments.

4.3.2 Third Spectral Moment

Theorem 4.3.3. *The 3rd spectral moment m_3 of P is*

$$m_3 = \mathbb{E}(\lambda^3) = 2\,\mathbb{E}(\Delta_i)\,\mathbb{E}(\frac{1}{d_h d_i d_j}),$$

where $\mathbb{E}(\Delta_i)$ *is the average number of triads a node is in and* $d_h d_i d_j$ *follows the joint degree distribution of triads* $p(d_h, d_i, d_j)$*: the probability that a triad is formed by nodes with degrees* d_h*,* d_i*, and* d_j*.*

Proof. As mentioned, the 3rd spectral moment of P can be viewed as the expected return probability of a 3-step random walk, which is equal to the summation of the return probability of a 3-step random walk starting from any node i divided by the number of nodes. Assume that nodes h, i, and j are connected to each other and form a triad. The triad will increase the overall return probability by $6\frac{1}{d_h d_i d_j}$ as it includes 6 closed walks: $h \to i \to j \to h$, $h \to j \to i \to h$, and so on. Denote Δ as the total number of triads in the graph, and there are $\Delta \cdot p(d_h, d_i, d_j)$ triads with nodes having degree d_h, d_i and d_j. Therefore,

$$m_3 = \mathbb{E}(\lambda^3) = \frac{\sum_{d_h,d_i,d_j} \frac{6}{d_h d_i d_j} \cdot \Delta \cdot p(d_h, d_i, d_j)}{n}. \tag{4.14}$$

By definition, $\mathbb{E}(\Delta_i) = \frac{3\Delta}{n}$, so

$$m_3 = 2\,\mathbb{E}(\Delta_i)\,\mathbb{E}(\frac{1}{d_h d_i d_j}). \tag{4.15}$$

□

4.3.3 Higher-Order Spectral Moments

The proof for Theorem 4.3.3 can be extended to a general case:

Theorem 4.3.4. *The* ℓ*-th spectral moment* m_ℓ *of* P *is*

$$m_\ell = \mathbb{E}(\lambda^\ell) = \mathbb{E}(CW_{\ell,i})\,\mathbb{E}(\frac{1}{d_1 d_2 \dots d_{\ell-1} d_\ell}),$$

where $\mathbb{E}(CW_{\ell,i})$ *denotes the average number of closed walks of length* ℓ *a node is in and*

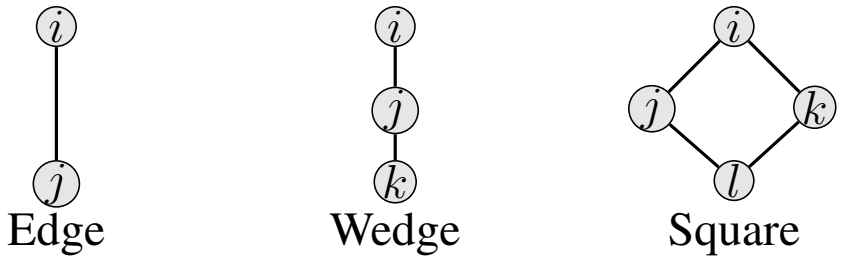


Figure 4.2: Structures related to the 4th spectral moment of P.

$d_1 d_2 \dots d_{\ell-1} d_\ell$ follows the joint degree distribution of closed walk of length ℓ formed by nodes with degrees d_1, d_2, …, d_ℓ.

Proof. The proof is straightforward:

$$\mathbb{E}(\lambda^\ell) = \frac{CW_\ell \, \mathbb{E}(\frac{1}{d_1 d_2 \dots d_{\ell-1} d_\ell})}{n} \tag{4.16}$$

$$= \mathbb{E}(CW_{\ell,i}) \, \mathbb{E}(\frac{1}{d_1 d_2 \dots d_{\ell-1} d_\ell}), \tag{4.17}$$

where CW_ℓ is the total number of closed walks of length ℓ. □

Basically, when ℓ is small, we can connect the ℓ-th spectral moment of P with the local structure of the graph. Next, we look at the case of $\ell = 4$.

Theorem 4.3.5. *The 4th spectral moment m_4 of P is*

$$m_4 = \mathbb{E}(\lambda^4) = (\, \mathbb{E}(d_i) + 4 \, \mathbb{E} \binom{d_i}{2} + 2 \, \mathbb{E}(\square_i) \,) \, \mathbb{E}(\frac{1}{d_i d_j d_k d_l}),$$

where $\mathbb{E}(d_i)$ is average degree, $\mathbb{E}(\square_i)$ is the average number of squares a node is in, and $d_i d_j d_k d_l$ follows the joint degree distribution of closed walks of length 4 formed by nodes with degrees d_i, d_j, d_k, d_l.

Proof. Figure 4.2 provides the three graph structures that result in closed walks of length 4: an edge, a wedge, and a square. Each edge contributes 2 closed walks; each wedge adds another 4 closed walks (without considering the walks that go through only one edge); similarly, each square contributes additional 8 closed walks. Let $|E|$ denote the number of

edges, ⁝ denote the number of wedges, and □ denote the number of squares in the graph. Hence,

$$\mathbb{E}(CW_{4,i}) = \frac{2|E|+4⁝+8□}{n}. \tag{4.18}$$

As $\mathbb{E}(d_i) = \frac{2|E|}{n}$, $⁝ = \sum_{d_i} \binom{d_i}{2} n_{d_i} = n\,\mathbb{E}\binom{d_i}{2}$ so $\frac{4⁝}{n} = 4\,\mathbb{E}\binom{d_i}{2}$, and $\mathbb{E}(□_i) = \frac{4□}{n}$, using Theorem 4.3.4, the theorem is proved. □

4.4 Relationship Between Spectral Moments and Network Properties

4.4.1 Degree Distribution

Degree distribution plays a vital role in network analysis. Different degree distributions have helped define different network families. Perhaps the most well-studied network family with connections to real-world networks are scale-free networks that exhibit a power law degree distribution, at least asymptotically [81, 82]. Degree distribution is often used to assess how well a network model fits real-world networks. Here, we look into the relationship between the spectral moments and the degree distribution, and we prove that the degree distribution helps bound the spectral moments.

Theorem 4.4.1. *The 2nd spectral moment m_2 of P is lower bounded by the degree distribution's expectation $\mathbb{E}(d_i)$ and variance $\mathrm{Var}(d_i)$:*

$$m_2 \geq \frac{\mathbb{E}^3(d_i)}{(\mathbb{E}^2(d_i) + \mathrm{Var}(d_i))^2}.$$

Proof. From Theorem 4.3.2, we have $m_2 = \mathbb{E}(d_i)\,\mathbb{E}(\frac{1}{d_i d_j})$. By Jensen's inequality, $\mathbb{E}(\frac{1}{d_i d_j}) \geq$

$\mathbb{E}^2(\frac{1}{\sqrt{d_i d_j}}) \geq \frac{1}{\mathbb{E}^2(\sqrt{d_i d_j})}$, and

$$\mathbb{E}(\sqrt{d_i d_j}) \leq \mathbb{E}(\frac{d_i + d_j}{2}) = \frac{\sum_{(i,j)\in E}(d_i + d_j)}{2|E|} \tag{4.19}$$

$$= \frac{\sum_i d_i^2}{2|E|} \quad \text{(as each } d_i \text{ gets counted } d_i \text{ times)} \tag{4.20}$$

$$= \frac{n\,\mathbb{E}(d_i^2)}{2|E|} = \frac{\mathbb{E}(d_i^2)}{\mathbb{E}(d_i)}. \tag{4.21}$$

Therefore, $\mathbb{E}(\frac{1}{d_i d_j}) \geq \frac{\mathbb{E}^2(d_i)}{\mathbb{E}^2(d_i^2)}$ and $m_2 \geq \frac{\mathbb{E}^3(d_i)}{\mathbb{E}^2(d_i^2)}$. As $\mathbb{E}(d_i^2) = \mathbb{E}^2(d_i) + \mathrm{Var}(d_i)$, $m_2 \geq \frac{\mathbb{E}^3(d_i)}{(\mathbb{E}^2(d_i)+\mathrm{Var}(d_i))^2}$. □

In Theorem 4.4.1, if $\mathrm{Var}(d_i) = 0$, the bound is tight: $m_2 = \frac{1}{\mathbb{E}(d_i)}$. The bound can be stated as $m_2 \geq \frac{\mathbb{E}^3(d_i)}{(\mathbb{E}^2(d_i)+\mathrm{Var}(d_i))^2} = \frac{1}{\mathbb{E}(d_i)+\frac{2\,\mathrm{Var}(d_i)}{\mathbb{E}(d_i)}+\frac{\mathrm{Var}^2(d_i)}{\mathbb{E}^3(d_i)}}$. Hence, given a fixed average degree $\mathbb{E}(d_i)$, a smaller degree variance $\mathrm{Var}(d_i)$ leads to a greater lower bound for m_2. In statistics, the value $\frac{\mathrm{Var}(d_i)}{\mathbb{E}(d_i)}$ is defined as the *index of dispersion* D for the degree distribution, so we can rewrite the bound as $m_2 \geq \frac{1}{\mathbb{E}(d_i)+2D+\frac{D^2}{\mathbb{E}(d_i)}}$.

4.4.2 Clustering Coefficient

The clustering coefficient helps measure the degree to which nodes in a graph tend to cluster together. Clustering coefficient is specifically used to analyze transitivity (triangles) in an undirected graph. Theorem 4.3.3 states that $m_3 = 2\,\mathbb{E}(\Delta_i)\,\mathbb{E}(\frac{1}{d_h d_i d_j})$, indicating that m_3 is closely related to the number of triangles. Next, we prove that m_3 and the degree distribution's expectation and variance ($\mathbb{E}(d_i)$ and $\mathrm{Var}(d_i)$) provide a lower bound for the global clustering coefficient C. Also, m_3 is upper bounded by C, $\mathbb{E}(d_i)$ and $\mathrm{Var}(d_i)$.

Corollary 4.4.1.1. *The 3rd moment m_3 is upper bounded by $\frac{1}{4}\,\mathbb{E}(\Delta_i)$.*

Proof. Because any node in a triangle has a degree of at least 2, $\mathbb{E}(\frac{1}{d_h d_i d_j}) \leq \frac{1}{8}$. Therefore, $m_3 = 2\,\mathbb{E}(\Delta_i)\,\mathbb{E}(\frac{1}{d_h d_i d_j}) \leq \frac{1}{4}\,\mathbb{E}(\Delta_i)$. □

Theorem 4.4.2. *Given the 3rd spectral moment m_3 of P, $\mathbb{E}(d_i)$ and $\mathrm{Var}(d_i)$, the global clustering coefficient C can be lower bounded as*

$$C \geq \frac{8m_3}{\mathbb{E}^2(d_i) - \mathbb{E}(d_i) + \mathrm{Var}(d_i)}.$$

Proof. Starting from the definition of clustering coefficient C:

$$C = \frac{|\text{Closed Paths of Length 2}|}{|\text{Paths of Length 2}|} = \frac{\Delta \times 6}{\sum_i 2\binom{d_i}{2}} = \frac{2n\,\mathbb{E}(\Delta_i)}{2n\,\mathbb{E}\binom{d_i}{2}} \tag{4.22}$$

$$= \frac{\mathbb{E}(\Delta_i)}{\mathbb{E}\binom{d_i}{2}} \tag{4.23}$$

Using Corollary 4.4.1.1, we get $m_3 \leq \frac{C}{4}\mathbb{E}\binom{d_i}{2}$, so $C \geq \frac{4m_3}{\mathbb{E}\binom{d_i}{2}}$. As $\mathbb{E}\binom{d_i}{2} = \frac{\mathbb{E}(d_i^2)-\mathbb{E}(d_i)}{2} = \frac{\mathbb{E}^2(d_i)-\mathbb{E}(d_i)+\mathrm{Var}(d_i)}{2}$, the proof is complete. □

Therefore, given fixed $\mathbb{E}(d_i)$ and $\mathrm{Var}(d_i)$, when m_3 is large, C has a greater lower bound. Also, m_3 can be upper bounded by C, $\mathbb{E}(d_i)$, and $\mathrm{Var}(d_i)$ using Theorem 4.4.2: $m_3 \leq \frac{C}{8}(\mathbb{E}(d_i)^2 - \mathbb{E}(d_i) + \mathrm{Var}(d_i))$. So, given fixed $\mathbb{E}(d_i)$ and $\mathrm{Var}(d_i)$, when the global clustering coefficient is small, m_3 has a smaller upper bound, which makes sense as the number of closed walk of length 3 relies on the number of triangles.

4.4.3 Connectivity

Because m_ℓ is the expected return probability of an ℓ-step random walk, it is equal to the fraction of walks of length ℓ that are closed. Hence, the graph is globally well-connected when m_ℓ is small, as it is more likely to have a walk which travels far away from the starting node. On the other hand, a large m_ℓ indicates that more walks are closed and it is difficult to travel from a node to another one which is far away. Here, we link the spectral moments to graph connectivity by extending the *Estrada index* [83]. The Estrada index of a graph G is defined as $EE(G) = \sum_{j=1}^{n} e^{\mu_j}$, where μ_j's are the eigenvalues of the

adjacency matrix A. As $EE(G) = \text{trace}(e^A) = \sum_{k=0}^{\infty} \frac{\text{trace}(A^k)}{k!}$, the Estrada index actually counts the number of closed walks, discounting longer walks; hence, it is sometimes used to measure the global connectivity of a graph. Therefore, we propose a variation of the Estrada index using the random walk transition matrix P, which we denote as $EE_P(G) = \sum_{j=1}^{n} e^{\lambda_j} = \sum_{j=1}^{n} \text{trace}(e^P) = \sum_{k=0}^{\infty} \frac{\text{trace}(P^k)}{k!} = n \sum_{k=0}^{\infty} \frac{m_k}{k!}$. Different from the Estrada index, $EE_P(G)$ computes the expected return probability of a random walk of any length, discounting longer walks. The smaller the $EE_P(G)$ value, the more well-connected the graph G.

4.4.4 Connected Components

In Section 4.4.3, we discussed the spectral moments and network connectivity. Here, we look into the relationship between the spectral moments of a network and those of its connected components.

Theorem 4.4.3. *Consider graph $G = (V, E)$ with k connected components $G_1, G_2, \ldots, G_{k-1}, G_k$. For each $G_i = (V_i, E_i)$, denote its ℓ-th spectral moment as $m_{i,\ell}$. Then, the ℓ-th spectral moment of G is the weighted average of $m_{i,\ell}$'s weighted by $|V_i|$'s, i.e., $m_\ell = \frac{\sum_i |V_i| m_{i,\ell}}{|V|}$.*

Proof. The theorem can be proved in two ways, both of which are straightforward. The first way is that one can view the transition matrix of the random walk on G as a block matrix where each block represents the transition matrix of a connected component. The second way is that a walk starting from node i cannot reach the nodes in other connected components, so the overall return probability of an ℓ-step random walk is the weighted sum of the expected return probability for each connected component. □

Table 4.1: Spectral Moments of Various Types of Graphs with n nodes ($n > 5$)

Graphs	$\mathbb{E}(d_i)$	$\mathbb{E}(\frac{1}{d_i d_j})$	**2nd moment** m_2	$\mathbb{E}(\Delta_i)$	$\mathbb{E}(\frac{1}{d_i d_j d_k})$	**3rd moment** m_3	$\mathbb{E}\binom{d_i}{2}$	$\mathbb{E}(\square_i)$	$\mathbb{E}(\frac{1}{d_i d_j d_k d_l})$	**4th moment** m_4
Complete Graphs	$n-1$	$\frac{1}{(n-1)^2}$	$\frac{1}{n-1}$	$\binom{n-1}{2}$	$\frac{1}{(n-1)^3}$	$\frac{n-2}{(n-1)^2}$	$\binom{n-1}{2}$	$3\binom{n-1}{3}$	$\frac{1}{(n-1)^4}$	$\frac{n^2-3n+3}{(n-1)^3}$
Cycles	2	$\frac{1}{4}$	$\frac{1}{2}$	0	NA	0	1	0	$\frac{1}{16}$	$\frac{3}{8}$
Star Graphs	$\frac{2(n-1)}{n}$	$\frac{1}{n-1}$	$\frac{2}{n}$	0	NA	0	$\frac{(n-1)(n-2)}{2n}$	0	$\frac{1}{(n-1)^2}$	$\frac{2}{n}$
Complete Bipartite Graph $K_{a,b}$	$\frac{2ab}{a+b}$	$\frac{1}{ab}$	$\frac{2}{a+b} = \frac{2}{n}$	0	NA	0	$\frac{ab(a+b-2)}{2(a+b)}$	$\frac{4\binom{a}{2}\binom{b}{2}}{a+b}$	$\frac{1}{a^2b^2}$	$\frac{2}{a+b} = \frac{2}{n}$
Wheels	$\frac{4(n-1)}{n}$	$\frac{n+2}{18(n-1)}$	$\frac{2(n+2)}{9n}$	$\frac{3(n-1)}{n}$	$\frac{1}{9(n-1)}$	$\frac{2}{3n}$	$\frac{(n-1)(n+4)}{2n}$	$\frac{4(n-1)}{n}$	$\frac{1}{27(n-1)}$	$\frac{2n+20}{27n}$

4.5 Representing Networks with Spectral Moments

The results in Sections 4.3 and 4.4 show that spectral moments of a network are closely related to its structure and various properties. As discussed, we propose to represent a graph as a point in the 3D space using its truncated spectral moments: (m_2, m_3, m_4), where we denote this point as the *spectral point* of the graph.

4.5.1 Spectral Points of Various Types of Graphs

In this section, we first explore the spectral points of various types of graphs including complete graphs, cycles, star graphs, complete bipartite graphs $K_{a,b}$, and wheels, which as we will show their spectral moments are functions of the number of nodes n. We derive the spectral moments for all such graphs. Table 4.1 lists the spectral moments of these graphs and the related information. We observe that (1) For complete graphs, the three spectral moments decreases when the number of nodes increases and when $n \to \infty$ they all converge to 0; (2) For cycles, the spectral point is fixed at (0.5, 0, 0.375), independent of n; (3) Star graphs and complete bipartite graphs share the same spectral moments, as star graphs are a special case of complete bipartite graph $K_{1,n-1}$. Their third spectral moment is 0 and the other two moments are $\frac{2}{n}$. (4) For wheels, the three spectral moments decreases when the number of nodes increases and when $n \to \infty$ they converge to $(\frac{2}{9}, 0, \frac{2}{27})$. We vary n from 100 to 10,000 and we plot the spectral points of these types of graphs in Figure 4.3. As expected, we find that (1) complete graphs, stars, and complete bipartite graphs are lines converging to (0, 0, 0); (2) wheels form a line which converges to $(\frac{2}{9}, 0, \frac{2}{27})$.

Next, we look into spectral points of the k-regular graphs ($k > 0$), where each node has

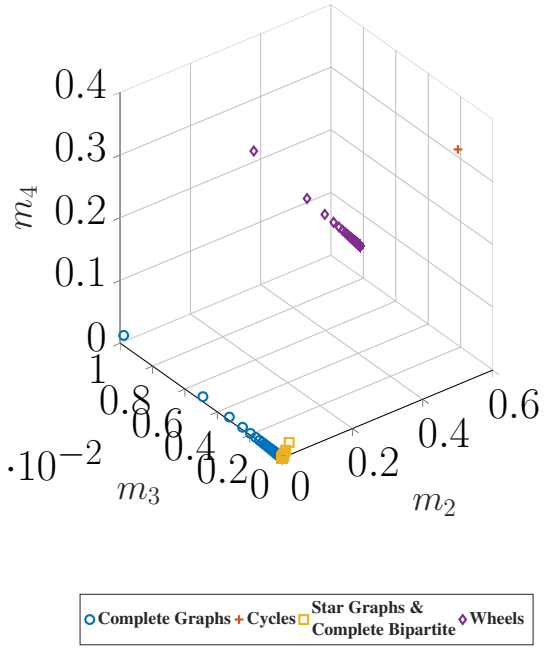


Figure 4.3: Spectral Moments of Various Types of Graphs

the same degree k. We find that the three spectral moments of k-regular graphs are upper bounded by $\frac{1}{k}$.

Theorem 4.5.1. *For a k-regular graph, (m_2, m_3, m_4) satisfies*

$$m_2 = \frac{1}{k}; \tag{4.24}$$

$$m_3 \leq \frac{k-1}{k^2} < \frac{1}{k}; \tag{4.25}$$

$$m_4 \leq \frac{1}{k}. \tag{4.26}$$

Proof. For m_2: as $\mathbb{E}(d_i) = k$ and $\mathbb{E}(\frac{1}{d_i d_j}) = \frac{1}{k^2}$, $m_2 = \frac{1}{k}$.

For m_3: clearly, $\mathbb{E}(\frac{1}{d_i d_j d_k}) = \frac{1}{k^3}$. As each node has k neighbors, it is in at most $\binom{k}{2}$ triangles, i.e., $\mathbb{E}(\Delta_i) \leq \binom{k}{2}$. Hence, $m_3 = 2\,\mathbb{E}(\Delta_i)\,\mathbb{E}(\frac{1}{d_i d_j d_k}) \leq \frac{k-1}{k^2} < \frac{1}{k}$.

For m_4: $\mathbb{E}(\frac{1}{d_i d_j d_k d_l}) = \frac{1}{k^4}$, and based on Theorem 4 in [84], the total number of closed walks of length 4, $CW_4 \leq nk^3$, so $\mathbb{E}(CW_{4,i}) \leq k^3$. Using Theorem 4.3.4, $m_4 \leq \frac{1}{k}$. □

Corollary 4.5.1.1. *The spectral points for all k-regular graphs with $k \geq 2$ are within the $[0, 0.5] \times [0, 0.5] \times [0, 0.5]$ cube.*

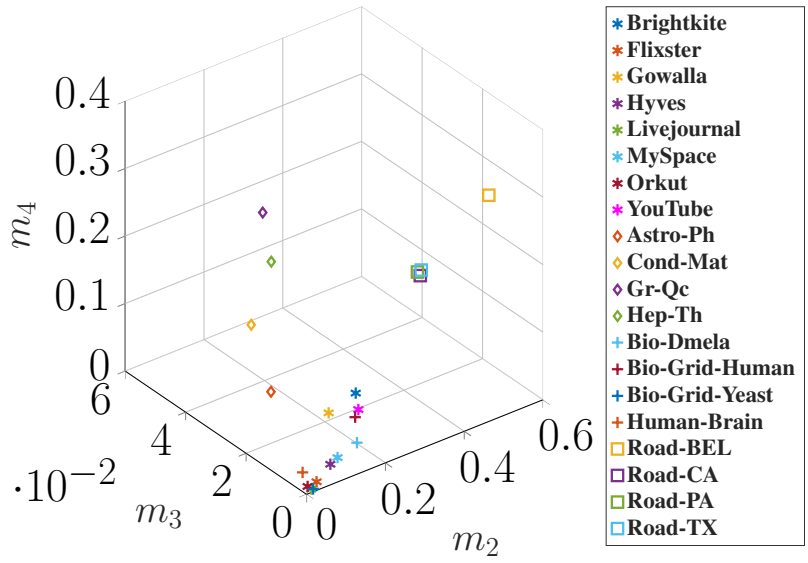


Figure 4.4: Spectral Points of Real-World Networks. Here, $*$'s are social networks, $\diamond$'s are collaboration networks, $\square$'s are road networks, and +'s are biological networks.

4.5.2 Representing Real-World Networks

Next, we move to real-world networks. We first introduce the datasets in our experiments.

Datasets

For our experiments, we use the same dataset in Section 3, which includes twenty real-world networks from four general network categories: social networks, collaboration networks, road networks, and biological networks.

Spectral Points of Real-World Networks

Figure 4.4 plots the spectral points of real-world networks. We observe the following for spectral moments of networks from different categories.

Social Networks. Social networks are often weakly scale free [81] with a log-normal degree distribution [85, 86] and exhibit a core-periphery structure [47, 87]. Hence, social networks have a relatively large degree variance, which makes the value m_2 generally smaller than that of road networks and collaboration networks. In general, social networks

have relatively small spectral moments, which shows they have better global connectivity. This observation accords to the fact that social networks exhibit the small-world phenomenon [88, 89] (e.g. in May 2011 the average path length between individuals in the Facebook graph was 4.7 [90]).

Collaboration Networks. Compared to networks from other categories, collaboration networks have greater m_3 values as the their clustering coefficient is much higher.

Road Networks. Compared to other categories, road networks have (1) a small degree variance, as often not many roads intersect at the same point; (2) more squares, as many parts of road networks resemble rectangular grids; (3) relatively low clustering coefficient, as triangles are uncommon. Due to these properties, road networks have large m_2 and m_4 values, but small m_3 values.

Biological Networks. Biological networks are often strongly scale free with a power-law degree distribution [91, 92] and a core-periphery structure [49]. In general, they share similar patterns with social networks.

4.5.3 Error Bound on Truncated Moments

Here, we provide an error bound on using the truncated spectral moments to represent a graph. We discuss the problem from two views: (I) the spectral distribution and (II) the network structure.

I. Spectral Distribution. Truncated spectral moments have been used to approximate the spectral density [28, 67]. Here, we provide a theoretical error bound on the maximum possible difference between the spectral distribution of two graphs that have the same truncated moments, i.e., worst-case scenario.

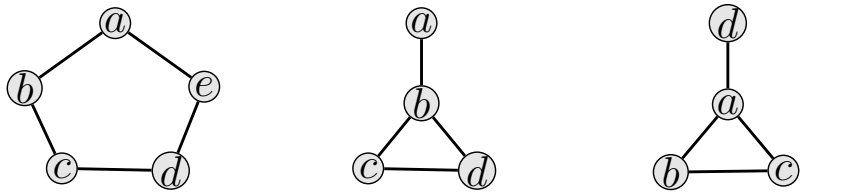
Lemma 4.5.2. *Given two graphs G_1 and G_2 with respective spectral density function p_1, p_2 of their random-walk transition matrices P_1 and P_2, if G_1 and G_2 have the same*

spectral points (m_2, m_3, m_4), then the Wasserstein distance $W_1(p_1, p_2)$ is upper bounded by $\frac{\pi}{4}$.

Proof. As we have discussed, both p_1 and p_2 are supported on $[0, 1]$. As $m_1 = 0$, p_1 and p_2 share the same first four moments. Based on Theorem 3 of [93], $W_1(p_1, p_2) \leq \frac{\pi}{4} + 3^4 \sum_{\ell=1}^{4} |m_\ell - m_\ell| = \frac{\pi}{4}$. □

Lemma 4.5.2 holds in the worst-case sense, as the first few moments can be good enough to determine the distribution with high accuracy. When one approximates the spectral density with the kernel polynomial method, only a few moments are needed by using smoothing techniques as the spectrum approximation error will decay exponentially [28, 94]. The empirical studies also show that recovery of spectrum of real-world networks using truncated moments performs much better than the theory would suggest [67].

II. Network Structure. As mentioned, m_ℓ represents the return probability of an ℓ-step random walk. We notice that a long closed walk (e.g. $k \geq 5$) can be a cycle of length k, but in more cases the closed walk is composed of multiple short closed walks. Let us see the cases of $k = 5$ in Figure 4.5. There are three cases which lead to a closed walk of length 5. Case 1: the starting node a is in a pentagon, so the pentagon will provide two closed walks of length 5; Case 2: The starting node a travels to b which is in a triangle, then b takes a closed walk of length 3, and finally, goes back to a; Case 3: a is in a triangle so it can have a close walk of length 3, and during this walk any node (a, b or c) takes a closed walk of 2. In both Cases 2 and 3, the closed walk of length 5 is decomposed into a closed walk of 2 and a closed walk of 3. In real-world networks, we find that most long closed walks belong to the latter situation, as higher-order structures like a long cycle is less frequent than lower-order structures such as edges, wedges, or triangles [95]. As m_2, m_3, m_4 can capture these short closed walks, we basically only lose the information on uncommon structures like long cycles.



Case 1: Pentagon | Case 2: Travel to a Triangle | Case 3: In a Triangle

Figure 4.5: Structures related to the m_5.

4.5.4 Time Complexity

In this chapter, we compute the accurate estimates of the low-order moments with the APPROXSPECTRALMOMENT algorithm proposed by [67]. The algorithm estimates the moments by simulating many random walks and computes the proportion of closed walks. To compute the ℓ-th spectral moment by simulating s random walks, it takes $O(s\ell)$ time. In our case, $\ell \leq 4$ and following the empirical results by [67], we set $s = 10,000$. For all networks used in our datasets, it takes only a few seconds to compute the three spectral moments. The Python code for computing moments has been released.[1]

4.6 Spectral Network Identification

In Section 4.5, we showed how networks can be represented and visualized using their truncated moments. Here, we demonstrate that spectral moments can be used for *network identification* [13], a problem which we will briefly review next.

4.6.1 Network Identification

Network identification [13] aims to identify the source from which an anonymized graph is sampled, to find the *identity* of a subgraph. Network identification can be formulated as follows: given a set of networks $N = \{N_1, N_2, \ldots, N_n\}$, and a subgraph G sampled from

[1] `https://github.com/shengminjin/EstimateSpectralMoments`

$N_i \in N$ using a sampling strategy S, we want to identify G, i.e., the network N_i from which G is sampled.

Based on the perturbation analysis of spectral density [28], the Wasserstein distance between the spectral density of a graph and the perturbed graph is bounded by the Frobenius norm of the perturbation. More specifically, suppose $\widetilde{A} = A + \Delta A$ is the perturbed graph matrix with spectral density $\widetilde{\mu}$, then $W_1(\mu, \widetilde{\mu}) \leq ||\Delta A||_F$. As one can view a subgraph as a result of a perturbation (removing nodes/edges) on the whole graph, the spectral moments of the subgraphs and the whole graphs should also be close. Therefore, we use spectral moments for network identification as they can capture the similarity between subgraphs and the whole graph. We use spectral moments as features directly to find the graph identities. Before detailing the experiments, we present the experimental setup.

4.6.2 Experimental Setup

From each real-world network, we sample many subgraphs representing graphs G which are to be identified. We vary the sampling proportion from 10% to 99% and sample using random node sampling. For each proportion, we sample two subgraphs. Hence, for each network we have $90 \times 2 = 180$ subgraphs, and for twenty networks, we have $180 \times 20 = 3,600$ samples to be identified.

4.6.3 Experiments

We use the spectral moments of each subgraph as its features and the name of the source networks as the class label, to train a multiclass classification model. We use 10-fold cross validation, and decision tree, SVM, k-NN and bagged trees as our classifiers. For spectral moments, we consider using the three spectral moments (m_2, m_3, m_4) and using the first 20 spectral moments. For evaluation, we provide the following two baselines for comparison.

Table 4.2: Network Identification with Spectral Moments

Type	Three Spectral Moments	First 20 Spectral Moments	Baselines	
			Top Eigenvalues	Random Prediction $(1/n)$
All Networks	82.0%	86.5%	62.4%	5%
Social Networks	95.5%	96.1%	74.8%	12.5%
Collaboration Networks	94.8%	97.1%	70.9%	25%
Road Networks	51.2%	53.3%	44.9%	25%
Biological Networks	99.7%	99.6%	90.4%	25%

1. **Top Eigenvalues**. Top eigenvalues have been used to study graph similarity [66]. We compute the top 5 eigenvalues of each sample as features for classification.

2. **Random Prediction**. A simple *random prediction*, so the accuracy will be $1/n$ where n is the number of networks.

We evaluate the methods for all networks and within each network category and report the performance for the best classifier in Table 4.2. The results show that spectral moments significantly outperform the baselines. The first 20 spectral moments perform best, but using three spectral moments one does not lose much predictive power, which confirms our discussion that the truncated spectral moments can keep most information on the spectral distribution and network structure of real-world networks.

4.7 Additional Related Work

Additionally, our work has links to the following areas:

I. 3D Network Embedding. Jin et al. [12] propose a 3D embedding method using Stochastic Kronecker Graph model. Their embedding method can capture the core-periphery structure of a network, and the embedding values can quantify the core strength of the network to some extent. Compared to their approach, spectral points carry further information on the network structure and properties.

II. Spectral Moments of other Associated Matrices. Preciado and colleagues [96, 97]

have provided detailed analysis on the spectral moments of the combinatorial Laplacian matrix ($D - A$) of graphs to connect the spectral moments with network structure. Here, we choose not to use the spectral moments of combinatorial Laplacian as the bounds on its eigenvalues are related to the size of the graph, and we prefer using the random walk transition matrix to have a compact embedding space.

III. Spectral Embedding. Recently, spectral information is used for different network embedding methods. One example is FGSD [98], which proposed a family of graph spectral distances that embeds the information as histograms and computes histograms on the biharmonic kernel of the graph. Another example is NetLSD [30], which computes and samples the heat or wave trace over the eigenvalues of a graph's normalized Laplacian to build embeddings. By using the spectral information, these embeddings are more focused on the predictive power, but are still relatively difficult to be interpreted.

4.8 Conclusion and Discussion

In this chapter, we introduce an approach to map all possible networks to a bounded 3D embedding space using the spectral moments of networks. We propose a 3D network embedding method, the *Spectral Point*, by using the truncated spectral moments of the network. We prove that spectral moments are closely related to network structure and various network properties. To the best of our knowledge, we are the first to study relationship between the spectral moments of the random walk transition matrix and network structure. We prove that the spectral moments are bounded by network properties such as the degree distribution and the global clustering coefficient, and spectral moments can be used as a measure for global connectivity. We prove the relationship between the spectral moments of the network and those of its connected components. We derive the spectral points of various types of graphs such as complete graphs, cycles, star graphs, complete bipartite graphs, and wheels. For k-regular graphs, we prove each dimension of the spectral points

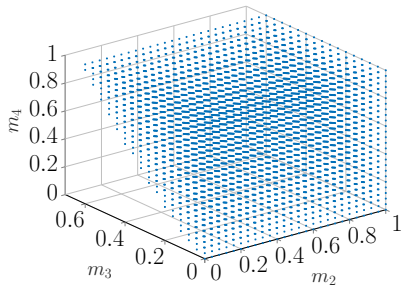


Figure 4.6: Feasible Spectral Embedding Space

are bounded by $\frac{1}{k}$. We analyze the spectral points of real-world graphs and show that their structure and properties identified in past literature are often captured by spectral points. Finally, we demonstrate that spectral moments can be used for network identification, i.e., to identify the source of an anonymized graph. The result shows that the spectral moments outperform the baselines and the truncated spectral moments do not lose much predictive power. We believe the spectral embedding space (i.e., the spectral zoo) can help obtain a better and quick understanding of the structure and properties of a network. Here, we further discuss some properties of the spectral embedding space.

Feasible Embedding Space for Spectral Moments. By definition, m_2, m_3 and m_4 are between 0 and 1. Hence, the whole spectral embedding space is a $1 \times 1 \times 1$ cube. However, not the whole cube is the feasible embedding space for networks due to the intrinsic relationship between these moments. For any random variable X with moments $m_j = \mathbb{E}(X^j)$, $m_3 \leq \sqrt{m_4 m_2 - m_2^3}$ and $m_3 \leq (\frac{4}{27})^{1/4} m_4^{3/4}$ [99]. Therefore, the embedding space which violates these conditions should be void. We plot the feasible spectral embedding space as the blue area in Figure 4.6.

Chapter 5

Network Shapes II: *Spectral Path*

5.1 Introduction

In Chapter 4, to represent a network with its spectral density, one can rely on its *spectral point*. However, there are a few drawbacks in using the spectral density (or spectral moments) as a representation of a graph: (1) in theory, different graphs can have the same spectral density. It is not difficult to construct such graphs. One way is to make a copy of a graph and take its union with the original graph. In this way, one doubles the size of the graph, but the spectral density does not change. Moreover, there exist non-isomorphic graphs sharing the same graph spectrum (i.e. *cospectral* or *isospectral* graphs) [100]; (2) the spectrum may dramatically change with a small change in graph structure [101].

To address these problems, we propose utilizing the spectral information of subgraphs. In this chapter, we propose to represent a graph using a 3D path in the spectral embedding space, which we denote as the *Spectral Path*. The spectral path connects the spectral moments of a network and its subgraphs. Figure 5.1 plots the spectral paths of a social network (YouTube) and a biological network. As we can see, though their spectral points of

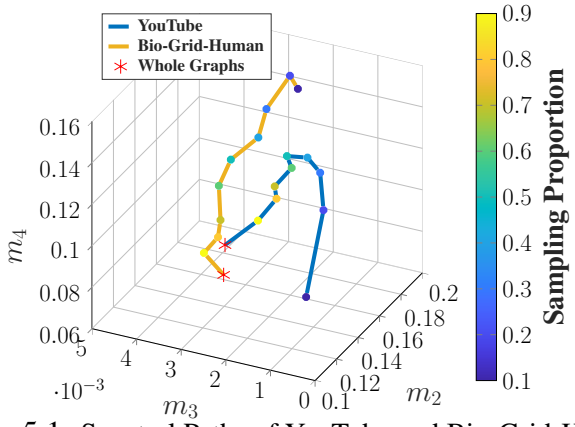
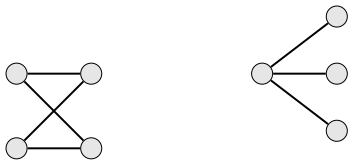


Figure 5.1: Spectral Paths of YouTube and Bio-Grid-Human

the whole networks (marked with *) are close, their spectral paths show different patterns.



Graph A Graph B

Figure 5.2: Cospectral Graphs

Our idea is inspired by the *Reconstruction Conjecture*, which is an open problem in theoretical computer science. The reconstruction conjecture — initially due to Kelly [102] and Ulam [102–104] — states that graphs are determined uniquely by their subgraphs. Bollobás has shown that the probability that a randomly chosen graph on n vertices is not reconstructible goes to 0 as $n \to \infty$ [35]. In other words, almost all graphs are reconstructible with their subgraphs. Naturally, by using the spectral density of subgraphs to capture subgraph information, one can better represent the whole graph. Figure 5.2 provides an example of two graphs that share the same spectral density but their subgraphs do not necessarily do so. In Figure 5.2, graphs G_A and G_B have the same graph spectrum of the random walk transition matrix: $[-1, 0, 0, 1]$. Hence, they also have the same spectral moments; for example, the second moment m_2 of their spectrum is $m_{2,A} = m_{2,B} = \frac{1}{2}$. If we randomly remove one node (and edges connected to it) from G_A, we get some subgraph $G_{A'}$. Subgraph $G_{A'}$ is 100% likely to be isomorphic to $\angle$, whose spectrum is $[-1, 0, 1]$ and its second spectral moment is $\frac{2}{3}$. However, if we randomly remove one node from G_B and get a subgraph $G_{B'}$, with 75% probability, we get $\angle$, and with 25% probability, we get an empty graph of 3 nodes whose spectrum is $[0, 0, 0]$ and its second spectral moment is 0. Therefore, though G_A and G_B are cospectral graphs, they have a different distribution of (second) spectral moments across all the subgraphs by randomly removing one node. If we take the expectation of the second spectral moment of $G_{A'}$ (or $G_{B'}$) over the distribution of the subgraphs, we get $\mathbb{E}(m_{2,A'}) = \frac{2}{3} \times 100\% = \frac{2}{3}$, but $\mathbb{E}(m_{2,B'}) = \frac{2}{3} \times 75\% + 0 \times 25\% = \frac{1}{2}$. In expectation, if we randomly remove one node, the second spectral moment of G_A will increase by $\frac{1}{6}$ but the second spectral moment of G_B will not change. In Section 5.5, we will provide a detailed theoretical analysis. These observations indicate that even if two networks have

similar spectral density, one can capture their difference in terms of substructures by using the expected spectral points (moments) of their subgraphs.

Overall, our contributions are mainly the following:

1. Spectral Path. We propose representing a network using its *Spectral Path*: a path connecting the spectral moments of the network and its subgraphs. Spectral paths provide an *interpretable-by-design* network representation.

2. Interpretability of Spectral Path. We study the interpretability of spectral paths by investigating the *shapes* of spectral paths. We provide the theoretical relationship between the spectral moments of a network and those of its subgraphs. We show how this relationship is closely related to the network structure. To the best of our knowledge, this work is the first to explore spectral moments of subgraphs and to study the relationship between spectral moments of subgraphs and those of the whole network.

3. Spectral Path Applications. We show that spectral path can be used for applications such as network visualization and network identification, i.e., identifying the source of an anonymized graph.

4. Spectral Paths of Cospectral Graphs. We theoretically explore the possibility of using the expected spectral moments of subgraphs to help distinguish cospectral graphs.

Organization. In Section 5.2, we present the algorithm to build a spectral path. In Section 5.3, we demonstrate the theoretical interpretation of spectral paths. We also present the relationship between the spectral moments of a network and its subgraphs, and how spectral paths are connected to the network structure. In Section 5.4, we show that spectral paths can be used for network visualization and network identification (that is, answering questions such as "Is this anonymized graph sampled from Twitter?"). Section 5.5 demonstrates how spectral paths provide a potential way to distinguish cospectral graphs. In Section 5.6, we provide more detailed proofs of some of the theorems in this chapter.

We review additional related work in Section 5.7 and conclude in Section 5.8.

5.2 Spectral Path

Here, we introduce the proposed network representation: the *spectral path* of a graph, which represents a network with the path connecting the spectral moments of a network and its subgraphs. The following simple steps can help build a spectral path:

Step 1: **Sample** many subgraphs from the network
Step 2: **Estimate** the expected spectral points using the spectral points of samples
Step 3: **Form** a spectral path by connecting the expected spectral points

The pseudocode is in Algorithm 2. The algorithm uses *Random Node Sampling* [37] to sample subgraphs from the network by (1) varying the proportion of nodes from 0% to 100% with step size s and (2) taking t independent samples for each proportion. We use Random Node Sampling as the subgraph can be viewed as a result of randomly removing nodes from a graph. For each sample and the whole network, the algorithm computes its spectral point (m_2, m_3, m_4). For all samples of the same size (sampling proportion p), the algorithm takes the average of their spectral points to estimate the expected spectral points. Hence, we get one (expected) spectral point for each sampling proportion p. Finally, it draws a path connecting the expected spectral points from $100\% \rightarrow \ldots \rightarrow 2p\% \rightarrow p\%$. Figure 5.3 illustrates the spectral path of YouTube with the spectral points of the samples. The figure shows that the spectral path can capture structural variations in subgraphs of different sizes.

Time Complexity. The majority of the computation time is dedicated to sampling subgraphs and computing three spectral moments for each subgraph. For one subgraph, random node sampling takes $\mathcal{O}(n+m)$ where $|V|= n$ and $|E|= m$. For large graphs, we com-

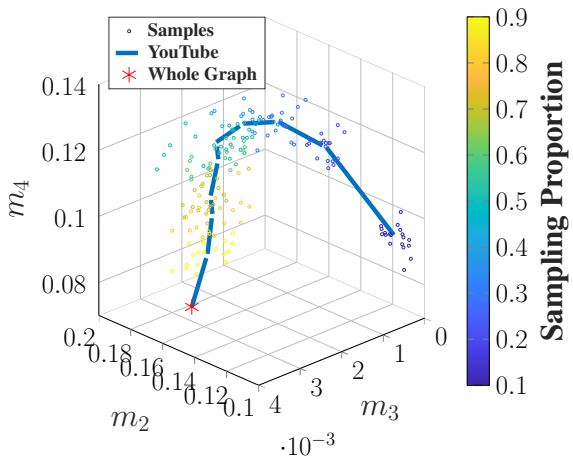


Figure 5.3: Spectral Path of YouTube and its sample points

pute accurate estimates of the low-order moments with the APPROXSPECTRALMOMENT algorithm [67]. The algorithm estimates the moments by simulating many random walks and computes the proportion of closed walks. To compute the ℓ-th spectral moment by simulating r random walks, it takes $O(r\ell)$ time. In our case, $\ell \leq 4$ and we set $r = 10,000$ following the empirical results of [67]. As the random walks can be taken in parallel, it only takes less than a few seconds to compute the three spectral moments even for large networks [15, 67]. Hence, for each subgraph, the time complexity is $\mathcal{O}((n+m)r\ell)$. We have a total of $\frac{100}{s} \times t + 1$ graphs (a network and its subgraphs) for which we compute spectral points. Thus, the time complexity for computing a spectral path is $\mathcal{O}(\frac{tr\ell}{s}(n+m))$, linear in the number of nodes and edges.

5.3 Interpretability of Spectral Paths

The interpretability of spectral path can be studied from two aspects (1) *the location of the spectral path in the 3D embedding space.* Here, we do not focus on this aspect. As mentioned, each component (dimension) of the spectral points is closely related to the network structure and network properties (see examples in [15]); (2) *the shape of the spectral paths*, which we will focus on here. The spectral points of different sampling sizes in the embedding space will determine the shape of a spectral path. Hence, we study

Algorithm 2 SPECTRAL PATH algorithm

input : an undirected network graph: $G(V, E)$
output : the Spectral Path of G: SP_G
parameter : s : sampling proportion step size;
t : number of samples for one proportion;
Expected_spectral_points = { };

for ($p = s;\ p < 100\%;\ p = p + s$) {
  Spectral_points = { }; %Spectral points for all the samples for proportion p
  for ($i = 1;\ i \leq t;\ i = i + 1$) {
    %Sample a p% subgraph G_p from G
    G_p = RandomNodeSampling(G, p);
    %Compute the spectral point of G_p (m_2, m_3, m_4)
    Spectral_point = ComputeMoments(G_P);
    Spectral_points.add(Spectral_point);
  }
  %Compute the average m_2, m_3, m_4 for all the samples for proportion p
  Expected_spectral_point = Average(Spectral_points);
  Expected_spectral_points.add(Expected_spectral_point);
}
%Compute the spectral point of G
Spectral_point = ComputeMoments(G);
Expected_spectral_points.add(Spectral_point);
%Form the spectral path of the (expected) spectral points, e.g., 100→90%...→10%
SP_G = Form_Path(Expected_spectral_points)
return SP_G;

the interpretability of spectral paths by investigating the relationship between the spectral points of subgraphs and that of the whole network. We start with the following questions:

$\mathcal{I}_1$) **Direction of the movement.** In which direction will the spectral point of a graph move if its nodes are randomly removed (equivalent to sampling a subgraph with random node sampling)? In other words, will spectral moments increase, decrease, or stay the same?;

$\mathcal{I}_2$) **Magnitude of the movement.** How far will the spectral point of a graph move under sampling?; and

$\mathcal{I}_3$) **Shape of spectral paths.** How do the shapes of spectral paths reveal the structural information of a network?

To answer these three questions ($\mathcal{I}_1$ to $\mathcal{I}_3$), we need to look into the spectral moments of subgraphs.

5.3.1 Second Spectral Moment of Subgraphs

We first look into the second spectral moment m_2 of subgraphs. In Theorem 5.3.1, we show what m_2 of a graph is expected to be when one node is removed, and the detailed proof is provided in Section 5.3.1. Then, we extend the result to removing k nodes ($k \geq 1$) from a graph in Theorem 5.3.4.

Theorem 5.3.1 (Expected second spectral moment m_2 of subgraphs after removing one node)**.** *In undirected graph $G = (V, E)$, where $|V_G| = n, |E_G| = m$, subgraph G' of G is obtained by removing one node from G uniformly at random. The expected second moment of G' is*

$$\mathbb{E}(m_{2,G'}) = m_{2,G} + \frac{2}{n(n-1)} \cdot \Big(\sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} - \sum_{\substack{(i,j)\in G \\ d_i=1, d_j=1}} \frac{1}{d_i d_j} + \delta_{\textit{triad}} \Big),$$

where d_i, d_j denote the degree of node i and j, and

$$\delta_{\textit{triad}} = \sum_{\substack{(i,j,k) \\ \textit{is a triad in } G}} \Big(\frac{1}{d_i d_j (d_i - 1)(d_j - 1)} + \frac{1}{d_i d_k (d_i - 1)(d_k - 1)} + \frac{1}{d_j d_k (d_j - 1)(d_k - 1)} \Big).$$

Especially, if G is a triangle-free graph, then $\mathbb{E}(m_{2,G'})$ reduces to:

$$\mathbb{E}(m_{2,G'}) = m_{2,G} + \frac{2}{n(n-1)} \cdot \Big(\sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} - \sum_{\substack{(i,j)\in G \\ d_i=1, d_j=1}} \frac{1}{d_i d_j} \Big)$$

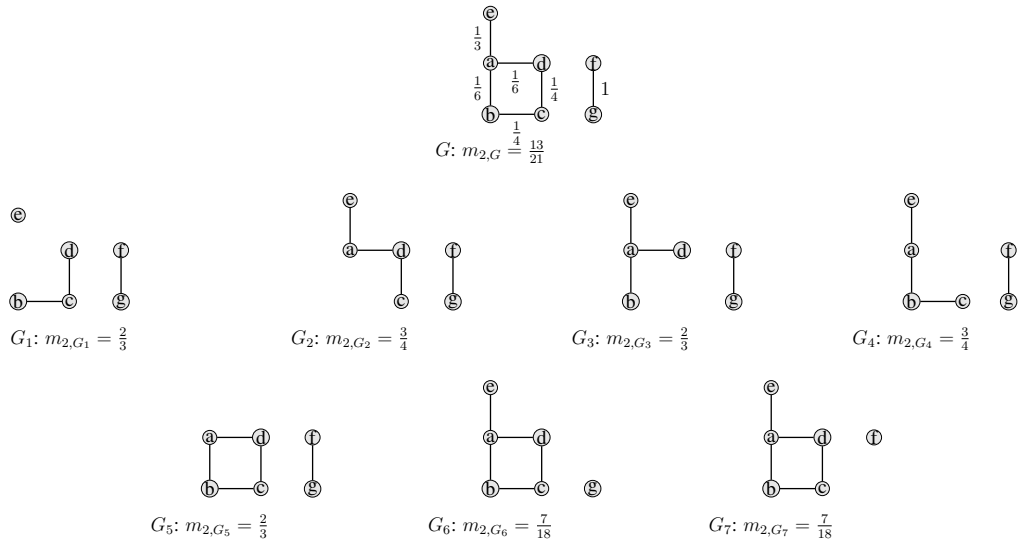


Figure 5.4: Example of Spectral Moment of Subgraphs

In Figure 5.4, we provide an example of a triangle-free graph to illustrate Theorem 5.3.1. We have a graph G with 7 nodes and 6 edges. The label on each edge is its value of $\frac{1}{d_i d_j}$. There are 7 subgraphs $(G_1, \ldots, G_7)$ by removing one node from G. Using Theorem 5.3.1, we get $\mathbb{E}(m_{2,G'}) = m_{2,G} + \frac{2}{n(n-1)} \cdot (\sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} - \sum_{\substack{(i,j)\in G \\ d_i=1, d_j=1}} \frac{1}{d_i d_j}) = \frac{13}{21} + \frac{2}{7\times 6}(\frac{1}{6} + \frac{1}{6} + \frac{1}{4} + \frac{1}{4} - 1) = \frac{11}{18}$, which is by definition equivalent to $\mathbb{E}(m_{2,G'}) = \frac{\sum_{i=1}^{7} m_{2,G_i}}{7} = [\frac{2}{3} + \frac{3}{4} + \frac{2}{3} + \frac{3}{4} + \frac{2}{3} + \frac{7}{18} + \frac{7}{18}]/7 = \frac{11}{18}$.

For an example of a graph with triangles, consider a complete graph $K_n (n > 3)$ as G. We get n subgraphs $(G_1, \ldots, G_n)$ by removing one node, each also being a complete graph K_{n-1}. We know that $m_{2,G} = \frac{1}{n-1}$, and using Theorem 5.3.1, we get $\mathbb{E}(m_{2,G'}) = m_{2,G} + \frac{2}{n(n-1)} \cdot (\sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} - \sum_{\substack{(i,j)\in G \\ d_i=1, d_j=1}} \frac{1}{d_i d_j} + \delta_{triad}) = \frac{1}{n-1} + \frac{2}{n(n-1)} \cdot (\binom{n}{2} \frac{1}{(n-1)^2} + \binom{n}{3} \frac{3}{(n-1)^2(n-2)^2}) = \frac{1}{n-2}$, which is the m_2 of K_{n-1}.

Interpretations of Theorem 5.3.1.

We provide interpretations for Theorem 5.3.1 by answering the aforementioned questions $\mathcal{I}_1$ to $\mathcal{I}_3$. In the proof of Theorem 5.3.1 (see Section 5.3.1), we partition any edge (i, j) of G into three types based on its end-points i and j's node degrees d_i and d_j:

► **Type I**: $d_i = 1, d_j = 1$;

► **Type II**: $d_i > 1, d_j = 1$;

► **Type III**: $d_i > 1, d_j > 1$.

Theorem 5.3.1 shows the expected m_2 of subgraphs when removing one node is closely related to frequencies of these three types of edges. Next, we will take triangle-free graphs as an example to show the interpretability of spectral paths, in terms of m_2. By removing one node from a triangle-free graph G, we get a subgraph G'. Compared to the second spectral moment of G, $m_{2,G'}$ is expected to add the term $\frac{2}{n(n-1)} \cdot (\sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} - \sum_{\substack{(i,j)\in G \\ d_i=1, d_j=1}} \frac{1}{d_i d_j})$.

$\mathcal{I}_1$) Direction of the movement of spectral point.

The direction of the movement of spectral point (moments) is basically the sign of the term, which is determined by the difference between the summation of $\frac{1}{d_i d_j}$ over Type III and Type I edges. If the difference is positive (or negative), then $m_{2,G'}$ is expected to increase (or decrease). Roughly speaking, if the graph G has more Type III edges and fewer Type I edges, $m_{2,G'}$ is expected to increase.

$\mathcal{I}_2$) Magnitude of the movement of spectral point.

Similarly, how far the spectral point will move is decided by the magnitude of the term. Besides the frequency of different types of edges, the size of a graph n also has an impact. When n is larger, the magnitude of the term is smaller, indicating that the more nodes G has, the smaller the impact on m_2 when one node is removed.

$\mathcal{I}_3$) Shape of spectral paths.

Assume that we start with a graph with many Type III edges. If we remove nodes one by one from it, it is very likely that (a) the second spectral moment will increase first, as there are still many Type III edges and the summation of $\frac{1}{d_i d_j}$ over them increases when the node degrees decrease; (b) if we keep removing more nodes, Type III edges get converted

to Type II or Type I, which makes the difference negative, and the second spectral moment starts to decrease. In other words, we will see a *turning point*; (c) finally, the moment will converge to 0 as more nodes are removed and the graph becomes an empty graph. In this case, the spectral path will show an *increasing-decreasing* pattern. Technically, it is possible for the turning point to happen in samples smaller than those taken to compute the spectral path. In that case, the spectral path will show an *increasing-only* pattern. Similarly, if the starting graph has mostly Type I edges, then the trend will be *decreasing-only*.

For general graphs with triangles, as theorem 5.3.1 shows, we need to consider an extra term δ_{triad} but in general they follow a similar pattern. These patterns will be observed in our later experiments.

Proof of Theorem 5.3.1.

To prove Theorems 5.3.1, we need Theorem 5.3.2 and Lemma 5.3.3.

Theorem 5.3.2 (Theorems 3.2, 3.3, 3.5 in [15])**.** *The 2nd, 3rd, and 4th spectral moments (m_2, m_3, m_4) of random walk transition matrix P are* $m_2 = \mathbb{E}(\lambda^2) = \mathbb{E}(d_i)\,\mathbb{E}(\frac{1}{d_i d_j})$, $m_3 = \mathbb{E}(\lambda^3) = 2\,\mathbb{E}(\Delta_i)\,\mathbb{E}(\frac{1}{d_h d_i d_j})$, *and* $m_4 = \mathbb{E}(\lambda^4) = [\,\mathbb{E}(d_i) + 4\,\mathbb{E}\binom{d_i}{2} + 2\,\mathbb{E}(\square_i)\,]\,\mathbb{E}(\frac{1}{d_i d_j d_k d_l})$, *where $\mathbb{E}(d_i)$ is the average degree, $d_i d_j$ follows the joint degree distribution $p(d_i, d_j)$, $\mathbb{E}(\Delta_i)$ is the average number of triads a node is in, $d_h d_i d_j$ follows the joint degree distribution of triads $p(d_h, d_i, d_j)$, $\mathbb{E}(\square_i)$ is the average number of squares a node is in, and $d_i d_j d_k d_l$ follows the joint degree distribution of closed walks of length 4 formed by nodes with degrees d_i, d_j, d_k, d_l.*

Note that in this theorem, for example, the term $\mathbb{E}(\frac{1}{d_i d_j})$ is the expected value of $\frac{1}{d_i d_j}$ over all edges, where d_i and d_j are the degree of the nodes connected by some edge.

Lemma 5.3.3. *Graph $H = (V, E)$ is a disjoint union of k graphs $G_1, G_2, \ldots, G_{k-1}, G_k$, i.e., $H = \bigcup_{i=1}^{k} G_i$. Let m_{ℓ,G_i} denote the ℓ-th spectral moment for $G_i = (V_i, E_i)$. Then,*

the ℓ-th spectral moment of H is the weighted average of m_{ℓ,G_i}'s weighted by $|V_i|$'s, i.e., $m_{\ell,H} = \frac{\sum_i |V_i| m_{\ell,G_i}}{|V|}$.

Proof. The lemma is a generalized version of Theorem 4.3 of [15], if we consider any graph as a disjoint union of its connected components. Therefore, the proof is similar. Note that one can view the transition matrix of the random walk on H as a block matrix where each block represents the transition matrix of some G_i. □

A special case of Lemma 5.3.3 is that if all graphs G_i have the same order, i.e., $|V_i| = c$, for some constant c, then $m_{\ell,H} = \frac{\sum_i m_{\ell,G_i}}{k}$.

Proof of Theorem 5.3.1.

Proof. As G has n nodes, we have n choices to remove only one node, so we can get n possible subgraphs (G'). We denote these subgraphs as $G_1, G_2, \ldots, G_n$, e.g., $G_1, G_2, \ldots, G_7$ in Figure 5.4. Here, we aim to get the expected spectral moment of G', which is $\mathbb{E}(m_{2,G'}) = \frac{\sum_i m_{2,G_i}}{n}$.

We use proof by construction. Construct a new graph $H = (V_H, E_H)$ as a disjoint union of all these subgraphs G_i's, i.e., $H = \bigcup_{i=1}^{n} G_i$. From Lemma 5.3.3, we have $m_{2,H} = \frac{\sum_i m_{2,G_i}}{n}$ as each G_i has $n-1$ nodes. Hence, deriving $\mathbb{E}(m_{2,G'})$ is equivalent to finding $m_{2,H}$. From Theorem 5.3.2, we have $m_{2,H} = \mathbb{E}_H(d_i)\,\mathbb{E}_H(\frac{1}{d_i d_j})$. For $\mathbb{E}_H(d_i)$,

$$\mathbb{E}_H(d_i) = \frac{2 \cdot |E_H|}{|V_H|} = \frac{2 \cdot (n-2)m}{n(n-1)}$$

(as each edge of G appears $n-2$ times in H, and G_i has $n-1$ nodes.)

$$= \frac{n-2}{n-1} \cdot \mathbb{E}_G(d_i). \tag{5.1}$$

To derive $\mathbb{E}_H(\frac{1}{d_i d_j})$, we compare it to $\mathbb{E}_G(\frac{1}{d_i d_j})$: for any edge (i,j) of G, there are $n-2$

copies in H, but if the removed node is a neighbor of i (or j) in G, the degree d_i (or d_j) will decrease which leads to an increase of $\frac{1}{d_i d_j}$. Therefore, $\mathbb{E}_H(\frac{1}{d_i d_j}) > \mathbb{E}_G(\frac{1}{d_i d_j})$. To get the exact increase quantity, we partition any edge (i, j) of G into three types based on its node degree:

► **Type I**: $d_i = 1, d_j = 1$. For such an edge, all of its $n - 2$ copies in H have $\frac{1}{d_i d_j} = 1$ as neither of i and j can lose other neighbors, so there will be no increment;

► **Type II**: $d_i > 1, d_j = 1$. For an edge of this type, if the removed node is a neighbor of i, then the edge contributes an increment $\frac{1}{(d_i-1)d_j} - \frac{1}{d_i d_j} = \frac{1}{d_i d_j (d_i-1)}$. Among the $n - 2$ copies in H, there are $d_i - 1$ such cases as each neighbor of i gets removed once, so the overall contribution is $(d_i - 1) \cdot \frac{1}{d_i d_j (d_i-1)} = \frac{1}{d_i d_j}$;

► **Type III**: $d_i > 1, d_j > 1$. Assume nodes i and j have c_{ij} common neighbors. If the removed node is a neighbor of i but not j, then the edge contributes an increment $\frac{1}{(d_i-1)d_j} - \frac{1}{d_i d_j} = \frac{1}{d_i d_j (d_i-1)}$. Among its $n-2$ copies in H, there are d_i-1-c_{ij} such cases (excluding j and the common neighbors), so the total contribution is $\frac{d_i-1-c_{ij}}{d_i d_j (d_i-1)}$. Similarly, if the removed node is a neighbor of j but not i, the total contribution for such cases is $\frac{d_j-1-c_{ij}}{d_i d_j (d_j-1)}$. If the removed node is a common neighbor of i and j, the increment is $\frac{1}{(d_i-1)(d_j-1)} - \frac{1}{d_i d_j} = \frac{1}{d_i d_j (d_i-1)} + \frac{1}{d_i d_j (d_j-1)} + \frac{1}{d_i d_j (d_i-1)(d_j-1)}$. As there are c_{ij} common neighbors, the contribution by such cases is $\frac{c_{ij}}{d_i d_j (d_i-1)} + \frac{c_{ij}}{d_i d_j (d_j-1)} + \frac{c_{ij}}{d_i d_j (d_i-1)(d_j-1)}$. Overall for one edge of Type III in G, it contributes the increment: $\frac{d_i-1-c_{ij}}{d_i d_j (d_i-1)} + \frac{d_j-1-c_{ij}}{d_i d_j (d_j-1)} + \frac{c_{ij}}{d_i d_j (d_i-1)} + \frac{c_{ij}}{d_i d_j (d_j-1)} + \frac{c_{ij}}{d_i d_j (d_i-1)(d_j-1)} = \frac{2}{d_i d_j} + \frac{c_{ij}}{d_i d_j (d_i-1)(d_j-1)}$.

Therefore, the total increment δ is $\delta = \sum_{\substack{(i,j)\in G \\ d_i>1, d_j=1}} \frac{1}{d_i d_j} + \sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} (\frac{2}{d_i d_j} + \frac{c_{ij}}{d_i d_j (d_i-1)(d_j-1)})$. We know that $\sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{c_{ij}}{d_i d_j (d_i-1)(d_j-1)} = \sum_{(i,j,k)\text{is a triad in } G} (\frac{1}{d_i d_j (d_i-1)(d_j-1)} + \frac{1}{d_i d_k (d_i-1)(d_k-1)} + \frac{1}{d_j d_k (d_j-1)(d_k-1)})$ and we denote it as δ_{triad}.

Thus, the total increment δ is $\delta = \sum_{\substack{(i,j)\in G \\ d_i>1, d_j=1}} \frac{1}{d_i d_j} + 2 \cdot \sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} + \delta_{\text{triad}}$; normalized by $|E_H|$, we get

$$\mathbb{E}_H(\frac{1}{d_i d_j}) = \mathbb{E}_G(\frac{1}{d_i d_j}) + \frac{\delta}{(n-2)m}. \tag{5.2}$$

Next, we compute $m_{2,H}$ using Equations 5.1 and 5.2:

$$\begin{aligned}
m_{2,H} &= \mathbb{E}_H(d_i)\,\mathbb{E}_H(\frac{1}{d_i d_j}) \\
&= \frac{n-2}{n-1}\,\mathbb{E}_G(d_i) \cdot (\mathbb{E}_G(\frac{1}{d_i d_j}) + \frac{\delta}{(n-2)m}) \\
&= \frac{n-2}{n-1}\,\mathbb{E}_G(d_i)\,\mathbb{E}_G(\frac{1}{d_i d_j}) + \frac{\delta \cdot \mathbb{E}_G(d_i)}{(n-1)m} \\
&= \mathbb{E}_G(d_i)\,\mathbb{E}_G(\frac{1}{d_i d_j}) + \frac{\delta \cdot \mathbb{E}_G(d_i)}{(n-1)m} - \frac{\mathbb{E}_G(d_i)\,\mathbb{E}_G(\frac{1}{d_i d_j})}{n-1} \\
&= m_{2,G} + \frac{\mathbb{E}_G(d_i)}{(n-1)m} \cdot (\delta - m \cdot \mathbb{E}_G(\frac{1}{d_i d_j})) \\
&= m_{2,G} + \frac{2}{n(n-1)} \cdot (\delta - m \cdot \mathbb{E}_G(\frac{1}{d_i d_j})) \quad (\text{as } \mathbb{E}_G(d_i) = \frac{2m}{n})
\end{aligned}$$

Note that $m \cdot \mathbb{E}_G(\frac{1}{d_i d_j}) = \sum_{(i,j)\in G} \frac{1}{d_i d_j}$, so we obtain the following which finalizes the proof:

$$\begin{aligned}
\delta - m \cdot \mathbb{E}_G(\frac{1}{d_i d_j}) &= \sum_{\substack{(i,j)\in G \\ d_i>1, d_j=1}} \frac{1}{d_i d_j} + 2 \cdot \sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} + \delta_{\text{triad}} - \sum_{(i,j)\in G} \frac{1}{d_i d_j} \\
&= \sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} - \sum_{\substack{(i,j)\in G \\ d_i=1, d_j=1}} \frac{1}{d_i d_j} + \delta_{\text{triad}},
\end{aligned}$$

If G is triangle-free, then $\delta_{\text{triad}} = 0$. The theorem is proved. □

A more general case.

Next, we explore a more general case, so we provide a bound for the expected m_2 by removing k nodes from a triangle-free graph in Theorem 5.3.4. When $k = 1$, the theorem is reduced to Theorem 5.3.1, so the bound is tight.

Theorem 5.3.4. *(Expected m_2 by Removing k Nodes) In undirected triangle-free graph $G = (V, E)$, where $|V_G| = n, |E_G| = m$, subgraph G' of G is obtained by removing k nodes from G uniformly at random. For $\mathbb{E}(m_{2,G'})$, the expected second moment of G', we have*

$$\mathbb{E}(m_{2,G'}) \leq m_{2,G} + \frac{2k}{n(n-1)} \cdot (\frac{n-1}{n-k} \cdot \sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} - \sum_{\substack{(i,j)\in G \\ d_i=1, d_j=1}} \frac{1}{d_i d_j}),$$

where d_i and d_j denote the degrees of node i and j, respectively.

For the detailed proof of Theorem 5.3.4, please refer to Section 5.6. We use the same triangle-free graph G in Figure 5.4, and we list all the 21 subgraphs by removing two nodes from G, in Figure 5.5. By Theorem 5.3.4, $\mathbb{E}(m_{2,G'}) \leq \frac{13}{21} + \frac{2\times2}{7\times6}(\frac{6}{5} \cdot (\frac{1}{6} + \frac{1}{6} + \frac{1}{4} + \frac{1}{4}) - 1) = \frac{13}{21} \approx 0.619$, while the actual $\mathbb{E}(m_{2,G'}) = \frac{\sum_{i=1}^{21} m_{2,G_i}}{21} \approx 0.594 < 0.619$.

5.3.2 Third and Fourth Spectral Moments

In this part, we provide two theorems for the expected third and fourth spectral moments of subgraphs ($\mathbb{E}(m_{3,G'})$ and $\mathbb{E}(m_{4,G'})$) when one node is removed. Theorem 5.3.5 provides an upper bound for $\mathbb{E}(m_{3,G'})$ indicating that whether $\mathbb{E}(m_{3,G'})$ is expected to increase or not over that of the original graph (m_3) depends on the weighted summation of $\frac{1}{d_i d_j d_k}$ over four different types of triads. In general, if there are more triads formed by higher degree nodes, $m_{3,G'}$ is expected to increase; if there are more triads with low degree nodes (i.e., with degree 2), $m_{3,G'}$ is expected to decrease. The forth moment m_4 is related to closed walks of length 4 (generated by edges, wedges, or squares). We provide a loose bound on $\mathbb{E}(m_{4,G'})$ in Theorem 5.3.6. Our experiments also show that m_4 has a high correlation with m_2. For detailed proofs, please refer to Section 5.6. In general, if we view triads or squares as higher-order edges of a network, a similar analysis on m_2 can be applied to m_3 and m_4, leading to increasing-decreasing, increasing-only, and decreasing-only patterns for these moments.

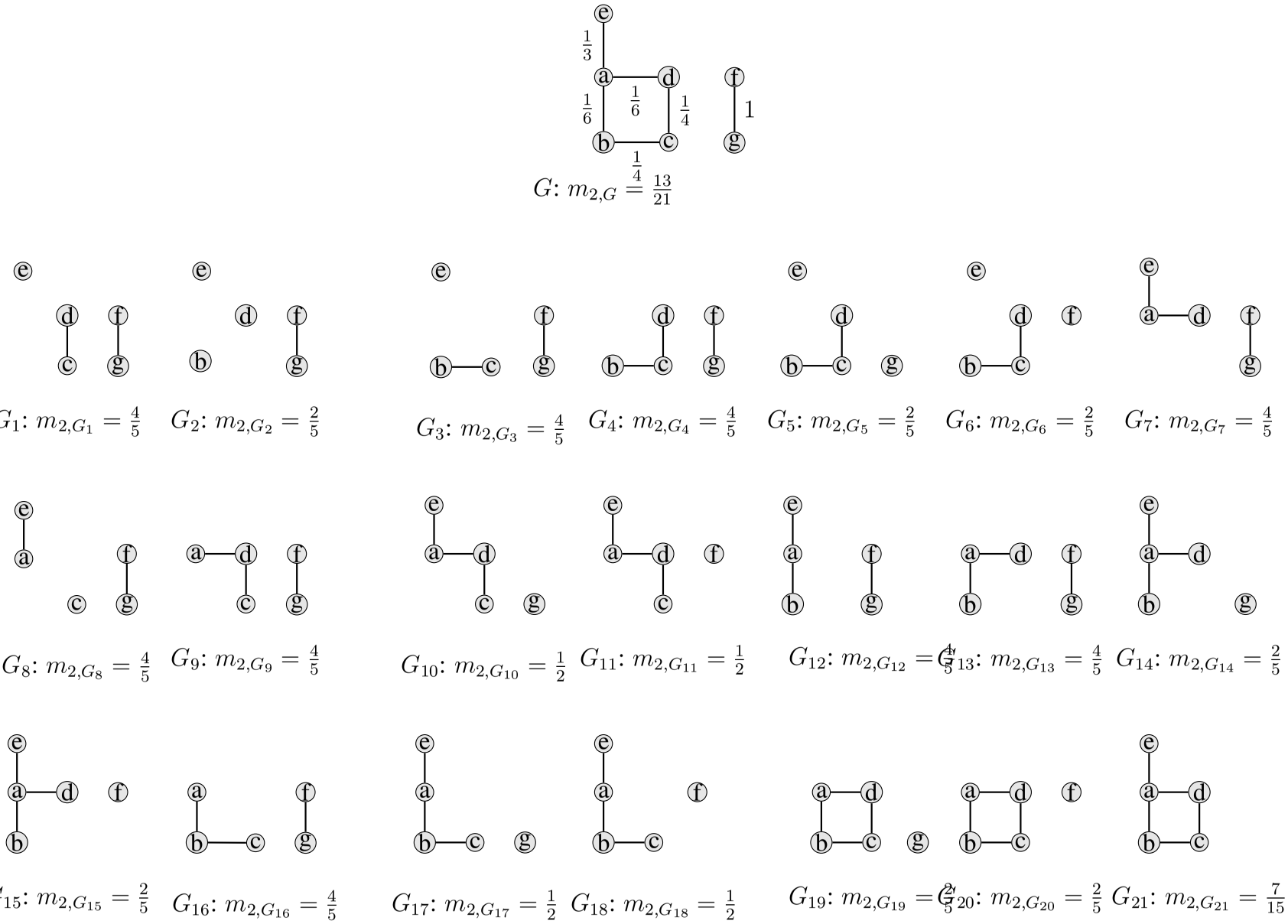


Figure 5.5: Example of Spectral Moment of Subgraphs (Removing $k = 2$ Nodes)

Theorem 5.3.5 (Expected third spectral moment m_3 of subgraphs after removing one node)**.** *In undirected graph $G = (V, E)$, where $|V_G| = n, |E_G| = m$, subgraph G' of G is obtained by removing one node from G uniformly at random. For $\mathbb{E}(m_{3,G'})$, the expected third moment of G', we have*

$$\mathbb{E}(m_{3,G'}) < m_{3,G} + \frac{6}{n(n-1)} \cdot (2 \sum_{\substack{(i,j,k)\in G \\ d_i>2, d_j>2, d_k>2}} \frac{1}{d_i d_j d_k} + \frac{1}{4} \sum_{\substack{(i,j,k)\in G \\ d_i>2, d_j>2, d_k=2}} \frac{1}{d_i d_j d_k} - \sum_{\substack{(i,j,k)\in G \\ d_i>2, d_j=2, d_k=2}} \frac{1}{d_i d_j d_k} - 2 \sum_{\substack{(i,j,k)\in G \\ d_i=2, d_j=2, d_k=2}} \frac{1}{d_i d_j d_k}).$$

Theorem 5.3.6 (Expected forth spectral moment m_4 of subgraphs after removing one node)**.** *In undirected graph $G = (V, E)$, where $|V_G| = n, |E_G| = m$, subgraph G' of G is obtained by removing one node from G uniformly at random. For $\mathbb{E}(m_{4,G'})$, the expected fourth moment of G', we have*

$$\mathbb{E}(m_{4,G'}) \leq \frac{16(n-2)}{n-1} m_{4,G}.$$

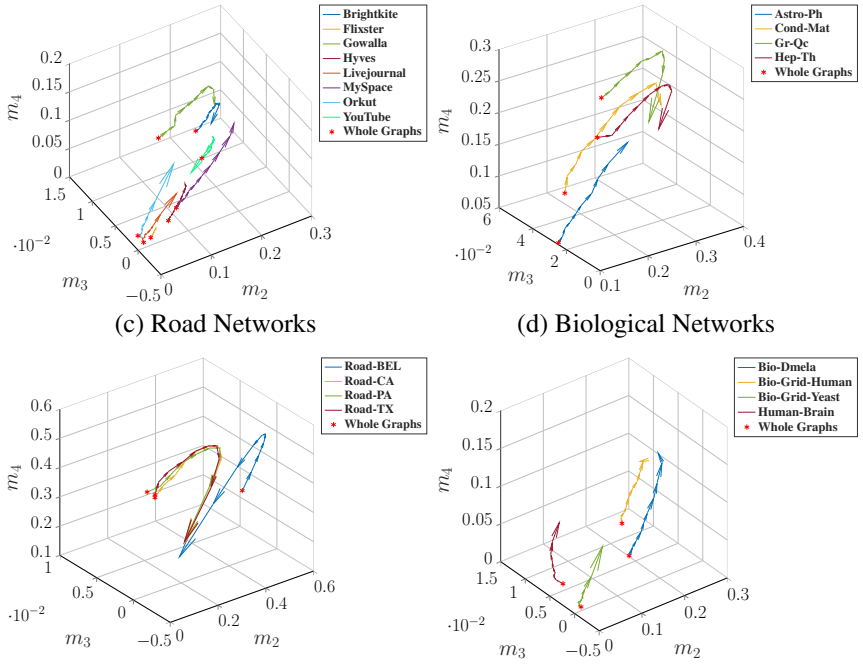
5.4 Applications

5.4.1 Experimental Setup

In our experiments, we generate spectral path for each network by varying the proportion of nodes from 0% to 100% with step size 10%, i.e., $s = 10\%$ in Algorithm 2; for each proportion (except for 100% which represents the whole graph), we generate 20 independently sampled subgraphs, i.e., $t = 20$ (for which we have a discussion in Section 5.8). In total, we generate $20 \times 9 + 1 = 181$ spectral points for each network, and we compute the expected spectral points for each sampling proportion. Hence, for each network, a spectral path connects 10 expected spectral points from (100% to 10%). Code and datasets are

Figure 5.6: Spectral Paths of Networks

(a) Social Networks (b) Collaboration Networks



(c) Road Networks (d) Biological Networks

publicly available.[1]

Datasets. We use the same dataset in Chapter 4 including 20 real-world networks from four network categories: social networks, collaboration networks, road networks, and biological networks.

5.4.2 Network Visualization

As a spectral path is a path connecting several 3D spectral points of a network and its subgraphs, we are able to plot them for visualization and to capture the network properties. In Figure 5.6, we plot the spectral paths of 20 real-world networks from four different categories. We have the following observations: (1) For m_2 and m_4, we see two common patterns: *increasing-decreasing*, and *increasing-only* which indicates the turning point happens before 10% samples of the graph. The observation shows that for most real-world graphs more edges are Type III edges; (2) among the eight social networks,

[1] `https://github.com/shengminjin/SpectralPath`

Brightkite, Gowalla, and YouTube show the increasing-decreasing pattern for both m_2 and m_4. It can be explained by their relatively low average degree and smaller graph size, so in small samples like 10% or 20%, most of the edges become Type I edges. Moreover, Brightkite and Gowalla show an increasing-decreasing trend on m_3 while Orkut shows an increasing-only trend on m_3, as Orkut has a much higher average degree so most of its triads are composed of high degree nodes. For the remaining networks, there is not much change on m_3 as they have a low clustering coefficient so they do not have as many triads as in those three; (3) for Collaboration networks, only Astro-Ph shows the increasing-only trend as it has a much higher average degree than other networks, so even 10% of the graph still has more edges among high-degree nodes. We observe the large change on m_3 for Cond-Mat, Gr-Qc and Hep-Th as in general collaboration networks have a high clustering coefficient and small samples of these sparse networks lose most triads or only have triads with low-degree nodes; (4) for road networks, we can see an early turning point (40%) on all m_2, m_3, and m_4. This can be explained by their low average degree so most edges and triads are among low degree nodes; (5) all biological networks show increasing-only trend as they all have a high average degree or high edge density. Overall, using the spectral path, one can get various insights on the graph structure.

5.4.3 Network Identification

Network identification [13] aims to identify the source network from which an anonymized graph is sampled, to find the *identity* of a subgraph. Network identification can be formulated as follows: given a set of networks $N = \{N_1, N_2, \ldots, N_n\}$, and a subgraph G sampled from $N_i \in N$ using a sampling strategy S, we want to identify G, i.e., the network N_i from which G is sampled. In the problem setting, there are a few assumptions: (1) The networks are not isomorphic, i.e., N_i and N_j are isomorphic $\implies i = j$, as isomorphic graphs are basically the same graph after anonymization; and (2) Subgraph G is not too small to lose its identity. It does not make much sense to verify the identity of a small

subgraph such as a triad, since it can be found in most networks.

Experimental Setup

From each of the 20 real-world networks, we sample many subgraphs representing graphs G which are to be identified. We vary the sampling proportion from 10% to 99% by using random node sampling with step size 1%. For each proportion, we sample two subgraphs. Hence, for each network, we have $90 \times 2 = 180$ subgraphs, and in total, $180 \times 20 = 3,600$ samples to be identified.

Experiments

As the spectral path has both spectral points of subgraphs and their relationship, we aim to explore whether one can improve the network identification performance by using the distances between an unidentified subgraph and the spectral path. For each subgraph, we compute the euclidean distances from its spectral point to the 10 expected spectral points of the spectral path, respectively. As there are 20 networks in total, for each subgraph we use the 10×20 = 200 distances as features and the name of the source networks as the class label, to train a multiclass classifier. We use 10-fold cross validation, and decision trees, SVM, k-NN, and bagged trees as our classifiers. For evaluation, we compare to the following four baselines: (1) **Three Spectral Moments**, where the spectral point (m_2, m_3, m_4) of each subgraph is used as features; (2) **First Twenty Spectral Moments**, where the first 20 spectral moments of each subgraph are used as features; (3) **KRONECKER HULL** [13], which uses Stochastic Kronecker Graph model to embed a network and its subgraphs and then a convex hull to represent the distribution of the embeddings. We use the distances between a subgraph to the convex hull of the whole network as features; (4) **GRAPH2VEC** is a graph embedding method which views a graph as a document and the rooted subgraphs around each node as words. It uses document embedding neural networks to embed a graph as a vector, which we use as a feature [68].

Table 5.1: Network Identification Accuracy with Spectral Path

Type	Spectral Path	Baselines			
		Three Spectral Moments	First 20 Spectral Moments	Kronecker Hull	GRAPH2VEC
All Networks	**96.3%**	82.0%	86.5%	84.4%	81.7%
Social Networks	**100.0%**	95.5%	96.1%	96.4%	83.7%
Collaboration Networks	**99.9%**	94.8%	97.1%	84.2%	97.4%
Road Networks	**86.8%**	51.2%	53.3%	76.8%	86.3%
Biological Networks	**100.0%**	99.7%	99.6%	80.4%	89.9%

We evaluate the methods for all networks and within each network category. We report the performance for the best classifier in Table 5.1, where spectral path significantly outperforms the baselines.

5.5 Spectral Path of Cospectral Graphs

Two non-isomorphic graphs are said to be cospectral with respect to a given matrix if they share the same graph spectrum. Well-known examples of cospectral graphs for the normalized Laplacian (as well as the random walk transition matrix) are complete bipartite graphs [78]. Butler et al. [105] propose constructing cospectral graphs by swapping in a bipartite subgraph with a *cospectral mate*. In general, cospectral graphs for the random walk transition matrix are related to the bipartite (sub)graphs.

Here, we aim to show that using the expected spectral moments (or spectral paths) of cospectral graphs may provide a potential way to distinguish two cospectral graphs. In this work, we use complete bipartite graphs as an example. It is known that the spectrum of a complete bipartite graph $K_{a,b}$ is $-1^{[1]}, 0^{[n-2]}, 1^{[1]}$, where $n = a + b$ and the exponent indicates multiplicity. Hence, its spectral moments are $m_i = 0$ for an odd i, and $m_i = \frac{2}{n}$ for an even i. It is easy to see that complete bipartite graphs of the same order are all cospectral, i.e., one can find another complete bipartite graph $K_{a',b'}$ where $a' + b' = a + b$. In Theorem 5.5.1, we prove that if one samples a subgraph from a complete bipartite graph using random node sampling, the expectation of its spectral moments are not only related

to n but also related to the values of a and b.

Theorem 5.5.1. *Given an undirected complete bipartite graph $G = (U, V, E)$ where $|U|= a$, $|V|= b$, $a + b = n$, and $a \leq b$, G' is a subgraph of G by removing k nodes from G uniformly at random. Then, when i is odd, $\mathbb{E}(m_{i,G'}) = 0$, and when i is even,*

$$\mathbb{E}(m_{i,G'}) = \begin{cases} \frac{2}{n-k} & k < a \\ \frac{2}{n-k}(1 - \frac{\binom{b}{k-a}}{\binom{n}{k}}) & a \leq k < b \\ \frac{2}{n-k}(1 - \frac{\binom{b}{k-a}+\binom{a}{k-b}}{\binom{n}{k}}) & k \geq b \end{cases}$$

Proof. As G' is a subgraph by removing k nodes from a complete bipartite graph G, G' is either a complete bipartite graph or an empty graph. Hence, $m_{i,G'}$ is always 0 when i is odd. When i is even, if $k < a$, G' is always a complete bipartite graph of $n - k$ nodes, so $\mathbb{E}(m_{i,G'}) = \frac{2}{n-k}$; if $a \leq k < b$, among all $\binom{n}{k}$ possible subgraphs, there are $\binom{b}{k-a}$ cases of G' being an empty graph where all the nodes in U are removed, and in the remaining cases, G' is a complete bipartite graph. Hence, $\mathbb{E}(m_{i,G'}) = \frac{\frac{2}{n-k}\cdot(\binom{n}{k}-\binom{b}{k-a})+0\cdot\binom{b}{k-a}}{\binom{n}{k}} = \frac{2}{n-k}(1 - \frac{\binom{b}{k-a}}{\binom{n}{k}})$; finally, if $k \geq b$, there are $\binom{b}{k-a}$ cases where all the nodes in U are removed, and $\binom{a}{k-b}$ cases when all the nodes in V are removed, where in both cases G' becomes an empty graph so we get $\mathbb{E}(m_{i,G'}) = \frac{2}{n-k}(1 - \frac{\binom{b}{k-a}+\binom{a}{k-b}}{\binom{n}{k}})$. □

Using Theorem 5.5.1, we get the following corollary:

Corollary 5.5.1.1. *Given two complete bipartite graphs $G_1 = (U_1, V_1, E_1)$ where $|U_1|= a_1, |V_1|= b_1$, and $G_2 = (U_2, V_2, E_2)$ where $|U_2|= a_2, |V_2|= b_2$, and $a_1 + b_1 = a_2 + b_2 = n$, $a_1 < a_2 \leq b_2 < b_1$. Let G_1' (or G_2') be a subgraph of G_1 (or G_2) by removing k nodes from*

G_1 (or G_2) uniformly at random. Then, for an even i, we have

$$\mathbb{E}(m_{i,G_1'} - m_{i,G_2'}) = \begin{cases} 0 & k < a_1 \\ -\frac{2}{n-k} \cdot \frac{\binom{b_1}{k-a_1}}{\binom{n}{k}} & a_1 \le k < a_2 \\ \frac{2}{n-k} \cdot \frac{\binom{b_2}{k-a_2} - \binom{b_1}{k-a_1}}{\binom{n}{k}} & a_2 \le k < b_2 \\ \frac{2}{n-k} \cdot \frac{\binom{b_2}{k-a_2} + \binom{a_2}{k-b_2} - \binom{b_1}{k-a_1}}{\binom{n}{k}} & b_2 \le k < b_1 \\ \frac{2}{n-k} \cdot \frac{\binom{b_2}{k-a_2} + \binom{a_2}{k-b_2} - \binom{b_1}{k-a_1} - \binom{a_1}{k-b_1}}{\binom{n}{k}} & k \ge b_1 \end{cases}$$

From Corollary 5.5.1.1, we notice that in general $\mathbb{E}(m_{i,G_1'} - m_{i,G_2'})$ is nonzero as long as $k \ge a_1$ (the special case is when $k = n$ or $n-1$, in which $\mathbb{E}(m_{i,G_1'} - m_{i,G_2'}) = 0$). Hence, if we remove more than a_1 nodes from two cospectral graphs (G_1, G_2) respectively, the expected (even) spectral moments of the corresponding subgraphs are different. Moreover, $a_1 < \frac{n}{2}$ as $a_1 + b_1 = a_2 + b_2 = n$, $a_1 < a_2 \le b_2 < b_1$. Hence, when $k \ge \frac{n}{2}$, $\mathbb{E}(m_{i,G_1'} - m_{i,G_2'})$ can be used to distinguish between two complete bipartite graphs. In other words, one can use random node sampling to sample subgraphs of less than $\frac{n}{2}$ nodes, and estimate $\mathbb{E}(m_{i,G_1'})$ and $\mathbb{E}(m_{i,G_2'})$, to distinguish two complete bipartite graphs. The idea can be extended to general cospectral graphs, as long as they have an explicit form of spectrum.

5.6 More Proofs

5.6.1 Proof of Theorem 5.3.4

Proof. As we are removing k nodes, there are $\binom{n}{k}$ possible subgraphs, denoted as $G_1, G_2, \ldots, G_{\binom{n}{k}}$. Each G_i has $n-k$ nodes. We construct $H = \bigcup_{i=1}^{\binom{n}{k}} G_i$. For each edge (i,j) in G, there will be $\binom{n-2}{k}$ copies in H when neither of the ending nodes is removed. Therefore,

$$E_H(d_i) = \frac{2 \cdot |E_H|}{|V_H|} = \frac{2m \cdot \binom{n-2}{k}}{(n-k)\binom{n}{k}} = \frac{2m(n-k-1)}{n(n-1)} = \frac{n-k-1}{n-1} \cdot \mathbb{E}_G(d_i). \tag{5.3}$$

Similarly, to get $E_H(\frac{1}{d_i d_j})$, we need to analyze the three types of edges. (1) **Type I**: $d_i = d_j = 1$. There will be no increment for edges of Type I; (2) **Type II**: $d_i > 1, d_j = 1$. If we consider x neighbors of node i are removed, the increment on the edge is $\frac{1}{(d_i-x)d_j} - \frac{1}{d_i d_j} = \frac{x}{(d_i-x)d_i d_j}$. Among the $\binom{n-2}{k}$ copies, there are $\binom{d_i-1}{x} \cdot \binom{n-2-(d_i-1)}{k-x}$ cases where x neighbors of i is removed. Moreover, x varies from 0 to $\min(d_i - 1, k)$, so the overall increment of an edge of Type II is $\sum_{x=0}^{\min(d_i-1,k)} \frac{x}{(d_i-x)d_i d_j}\binom{d_i-1}{x} \cdot \binom{n-2-(d_i-1)}{k-x} = \frac{1}{d_i d_j} \sum_{x=1}^{\min(d_i-1,k)} \binom{d_i-1}{x-1} \cdot \binom{n-2-(d_i-1)}{k-x} \leq \frac{1}{d_i d_j} \cdot \binom{n-2}{k-1}$ (as $\frac{x}{d_i-x}\binom{d_i-1}{x} = \binom{d_i-1}{x-1}$ and the bound is tight when $k \leq (d_i - 1)$); (3) **Type III**: $d_i > 1, d_j > 1$. Assume x neighbors of i and y neighbors of j are removed, then the increment is $\frac{1}{(d_i-x)(d_j-y)} - \frac{1}{d_i d_j} = \frac{x}{(d_i-x)d_i d_j} + \frac{y}{(d_j-y)d_i d_j} + \frac{xy}{(d_i-x)(d_j-y)d_i d_j}$. As G is a triangle-free graph, H is also triangle-free, and i and j have no common neighbors. Therefore, among the $\binom{n-2}{k}$ copies, there are $\binom{d_i-1}{x}\binom{d_j-1}{y}\binom{n-d_i-d_j}{k-x-y}$ cases when x neighbors of i and y neighbors of j are removed. Moreover, x varies from 0 to $\min(d_i - 1, k)$, y varies from 0 to $\min(d_j - 1, k)$ and $x + y \leq k$, so the overall increment of an edge of Type III is $\sum_{x=0}^{\min(d_i-1,k)} \sum_{y=0}^{\min(d_j-1,k-x)} \binom{d_i-1}{x}\binom{d_j-1}{y}\binom{n-d_i-d_j}{k-x-y}(\frac{x}{(d_i-x)d_i d_j} + \frac{y}{(d_j-y)d_i d_j} + \frac{xy}{(d_i-x)(d_j-y)d_i d_j})$. Note that

$$
\begin{aligned}
&\sum_{x=0}^{\min(d_i-1,k)} \sum_{y=0}^{\min(d_j-1,k-x)} \binom{d_i-1}{x}\binom{d_j-1}{y}\binom{n-d_i-d_j}{k-x-y}\frac{x}{(d_i-x)d_i d_j} \\
&= \sum_{x=0}^{\min(d_i-1,k)} \binom{d_i-1}{x}\frac{x}{(d_i-x)d_i d_j} \cdot \sum_{y=0}^{\min(d_j-1,k-x)} \binom{d_j-1}{y}\binom{n-d_i-d_j}{k-x-y} \\
&= \sum_{x=1}^{\min(d_i-1,k)} \binom{d_i-1}{x-1}\frac{1}{d_i d_j} \cdot \binom{n-d_i-1}{k-x} \\
&\leq \frac{1}{d_i d_j} \cdot \binom{n-2}{k-1}; \text{(the bound is tight when } k \leq (d_i-1)\text{)}.
\end{aligned}
$$

Due to the symmetry, the summation over $\frac{y}{(d_j-y)d_i d_j}$ is also less or equal to $\frac{1}{d_i d_j} \cdot \binom{n-2}{k-1}$. For the summation over $\frac{xy}{(d_i-x)(d_j-y)d_i d_j}$:

$$\sum_{x=0}^{\min(d_i-1,k)} \sum_{y=0}^{\min(d_j-1,k-x)} \binom{d_i-1}{x}\binom{d_j-1}{y}\binom{n-d_i-d_j}{k-x-y}\frac{xy}{(d_i-x)(d_j-y)d_id_j}$$
$$= \sum_{x=1}^{\min(d_i-1,k)} \sum_{y=1}^{\min(d_j-1,k-x)} \binom{d_i-1}{x-1}\binom{d_j-1}{y-1}\binom{n-d_i-d_j}{k-x-y}\frac{1}{d_id_j} \le \frac{1}{d_id_j}\cdot\binom{n-2}{k-2}$$

The increment of an edge of Type III is $\le \frac{1}{d_id_j}\cdot(2\binom{n-2}{k-1}+\binom{n-2}{k-2})$, so the total increment over all the edges is less or equal to $\delta = \binom{n-2}{k-1}\cdot \sum_{\substack{(i,j)\in G\\ d_i>1,d_j=1}} \frac{1}{d_id_j} + (2\binom{n-2}{k-1}+\binom{n-2}{k-2})\cdot \sum_{\substack{(i,j)\in G\\ d_i>1,d_j>1}} \frac{1}{d_id_j}$, and we normalize it by $|E_H|$:

$$\begin{aligned}
\frac{\delta}{|E_H|} &= \frac{\binom{n-2}{k-1}\cdot \sum_{\substack{(i,j)\in G\\ d_i>1,d_j=1}} \frac{1}{d_id_j} + (2\binom{n-2}{k-1}+\binom{n-2}{k-2})\cdot \sum_{\substack{(i,j)\in G\\ d_i>1,d_j>1}} \frac{1}{d_id_j}}{\binom{n-2}{k}m} \\
&= \frac{k}{(n-k-1)m}\cdot \sum_{\substack{(i,j)\in G\\ d_i>1,d_j=1}} \frac{1}{d_id_j} \\
&+ (\frac{2k}{(n-k-1)m} + \frac{k(k-1)}{(n-k)(n-k-1)m})\cdot \sum_{\substack{(i,j)\in G\\ d_i>1,d_j>1}} \frac{1}{d_id_j} \\
&= \frac{k}{(n-k-1)m}\cdot(\sum_{\substack{(i,j)\in G\\ d_i>1,d_j=1}} \frac{1}{d_id_j} + (2+\frac{k-1}{n-k}) \sum_{\substack{(i,j)\in G\\ d_i>1,d_j>1}} \frac{1}{d_id_j})
\end{aligned}$$

For simplicity, we denote $\delta' = \sum_{\substack{(i,j)\in G\\ d_i>1,d_j=1}} \frac{1}{d_id_j} + (2+\frac{k-1}{n-k})\cdot \sum_{\substack{(i,j)\in G\\ d_i>1,d_j>1}} \frac{1}{d_id_j}$, and we get:

$$\mathbb{E}_H(\frac{1}{d_id_j}) \le \mathbb{E}_G(\frac{1}{d_id_j}) + \frac{k}{(n-k-1)m}\delta' \tag{5.4}$$

Next, we compute $m_{2,H}$ using Equations 5.3 and 5.4:

$$\begin{aligned}
m_{2,H} &= \mathbb{E}_H(d_i)\,\mathbb{E}_H(\frac{1}{d_id_j}) \\
&\le \frac{n-k-1}{n-1}\,\mathbb{E}_G(d_i)\cdot(\mathbb{E}_G(\frac{1}{d_id_j}) + \frac{k}{(n-k-1)m}\delta')
\end{aligned}$$

$$
\begin{aligned}
&= \frac{n-k-1}{n-1}\,\mathbb{E}_G(d_i)\,\mathbb{E}_G(\frac{1}{d_i d_j}) + \frac{k\delta' \cdot \mathbb{E}_G(d_i)}{(n-1)m} \\
&= \mathbb{E}_G(d_i)\,\mathbb{E}_G(\frac{1}{d_i d_j}) + \frac{k\delta' \cdot \mathbb{E}_G(d_i)}{(n-1)m} - \frac{k\,\mathbb{E}_G(d_i)\,\mathbb{E}_G(\frac{1}{d_i d_j})}{n-1} \\
&= m_{2,G} + \frac{k\,\mathbb{E}_G(d_i)}{(n-1)m} \cdot (\delta' - m \cdot \mathbb{E}_G(\frac{1}{d_i d_j})) \\
&= m_{2,G} + \frac{2k}{n(n-1)} \cdot (\delta' - m \cdot \mathbb{E}_G(\frac{1}{d_i d_j})) \quad (\text{as } \mathbb{E}_G(d_i) = \frac{2m}{n})
\end{aligned}
$$

Note that $m \cdot \mathbb{E}_G(\frac{1}{d_i d_j}) = \sum_{(i,j)\in G} \frac{1}{d_i d_j}$, so

$$
\begin{aligned}
\delta' - m \cdot \mathbb{E}_G(\frac{1}{d_i d_j}) &= \sum_{\substack{(i,j)\in G \\ d_i>1, d_j=1}} \frac{1}{d_i d_j} + (2 + \frac{k-1}{n-k}) \cdot \sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} - \sum_{(i,j)\in G} \frac{1}{d_i d_j} \\
&= \frac{n-1}{n-k} \cdot \sum_{\substack{(i,j)\in G \\ d_i>1, d_j>1}} \frac{1}{d_i d_j} - \sum_{\substack{(i,j)\in G \\ d_i=1, d_j=1}} \frac{1}{d_i d_j},
\end{aligned}
$$

which finalizes the proof. □

5.6.2 Proof of Theorem 5.3.5

Proof. The idea of the proof is similar to that of Theorem 5.3.1. There are n subgraphs $G_1, G_2, \ldots, G_n$. We construct $H = \bigcup_{i=1}^{n} G_i$, so $m_{3,H} = \frac{\sum_i m_{3,G_i}}{n} = \mathbb{E}(m_{3,G'})$. From Theorem 5.3.2, $m_3 = \mathbb{E}(\lambda^3) = 2\,\mathbb{E}(\Delta_i)\,\mathbb{E}(\frac{1}{d_i d_j d_k})$. In the formula, $\mathbb{E}(\Delta_i)$ is the average number of triads a node is in, and by definition $\mathbb{E}(\Delta_i) = \frac{3\Delta}{n}$, where Δ is the number of triads of a graph. $\mathbb{E}(\frac{1}{d_i d_j d_k})$ is the expected value of $\frac{1}{d_i d_j d_k}$ over all triads. Hence, $m_{3,H} = 2\,\mathbb{E}_H(\Delta_i)\,\mathbb{E}_H(\frac{1}{d_i d_j d_k})$. For any triad (i,j,k) in G, it has $n-3$ copies in which no member of i, j, k is removed, so $\mathbb{E}_H(\Delta_i) = \frac{3\Delta_H}{|V_H|} = \frac{3(n-3)\Delta_G}{n(n-1)} = \frac{n-3}{n-1}\,\mathbb{E}_G(\Delta_i)$.

To get $\mathbb{E}_H(\frac{1}{d_i d_j d_k})$, we consider four types of triads based on their node degrees.

► **Type I**: $d_i = 2, d_j = 2, d_k = 2$. For such a triad, all of its $n-3$ copies have $\frac{1}{d_i d_j d_k} = \frac{1}{8}$ as neither of i and j can lose other neighbors, so there will be no increment;

► **Type II**: $d_i > 2, d_j = 2, d_k = 2$. For a triad of this type, if the removed node is a neighbor of i, then the triad contributes an increment $\frac{1}{(d_i-1)d_jd_k} - \frac{1}{d_id_jd_k} = \frac{1}{d_id_jd_k(d_i-1)}$. Among the $n-3$ copies, there are $d_i - 2$ such cases, so the overall contribution is $(d_i - 2) \cdot \frac{1}{d_id_jd_k(d_i-1)} = \frac{1}{d_id_jd_k} \cdot \frac{d_i-2}{d_i-1} < \frac{1}{d_id_jd_k}$.

► **Type III**: $d_i > 2, d_j > 2, d_k = 2$. If the removed node is a neighbor of i but not connected to j (or a neighbor of j but not connected to i) and not including k, the triad contributes an increment $\frac{1}{(d_i-1)d_jd_k} - \frac{1}{d_id_jd_k} = \frac{1}{d_id_jd_k(d_i-1)}$ (or $\frac{1}{d_id_jd_k(d_j-1)}$). If the removed node is a common neighbor of i and j, the increment is $\frac{1}{(d_i-1)(d_j-1)d_k} - \frac{1}{d_id_jd_k} = \frac{1}{d_id_jd_k(d_i-1)} + \frac{1}{d_id_jd_k(d_j-1)} + \frac{1}{d_id_jd_k(d_i-1)(d_j-1)}$. Assume i and j have c_{ij} common neighbors not including k, then the overall increment of a triad of Type III is $(d_i - 2 - c_{ij}) \cdot \frac{1}{d_id_jd_k(d_i-1)} + (d_j - 2 - c_{ij}) \cdot \frac{1}{d_id_jd_k(d_j-1)} + c_{ij} \cdot (\frac{1}{d_id_jd_k(d_i-1)} + \frac{1}{d_id_jd_k(d_j-1)} + \frac{1}{d_id_jd_k(d_i-1)(d_j-1)}) = \frac{1}{d_id_jd_k}(\frac{d_i-2}{d_i-1} + \frac{d_j-2}{d_j-1} + \frac{c_{ij}}{(d_i-1)(d_j-1)})$. Note that c varies from 0 to $\min(d_i, d_j) - 2$, so the overall increment is less than $\frac{9}{4} \cdot \frac{1}{d_id_jd_k}$.

► **Type IV**: $d_i > 2, d_j > 2, d_k > 2$. If the removed node is a neighbor of i but not connected to j or k, the increment is $\frac{1}{d_id_jd_k} \cdot \frac{1}{d_i-1}$; if the removed node is a common neighbor of i and j but not connected to k, then the increment is $\frac{1}{d_id_jd_k} \cdot (\frac{1}{d_i-1} + \frac{1}{d_j-1} + \frac{1}{(d_i-1)(d_j-1)})$; if the removed node is a common neighbor of i, j and k, the increment is $\frac{1}{(d_i-1)(d_j-1)(d_k-1)} - \frac{1}{d_id_jd_k} = \frac{1}{d_id_jd_k} \cdot (\frac{1}{d_i-1} + \frac{1}{d_j-1} + \frac{1}{d_k-1} + \frac{1}{(d_i-1)(d_j-1)} + \frac{1}{(d_i-1)(d_k-1)} + \frac{1}{(d_j-1)(d_k-1)} + \frac{1}{(d_i-1)(d_j-1)(d_k-1)})$. Similarly, we can get the overall increment for a triad of Type IV: $\frac{1}{d_id_jd_k} \cdot (\frac{d_i-2}{d_i-1} + \frac{d_j-2}{d_j-1} + \frac{d_k-2}{d_k-1} + \frac{c_{ij}}{(d_i-1)(d_j-1)} + \frac{c_{ik}}{(d_i-1)(d_k-1)} + \frac{c_{jk}}{(d_j-1)(d_k-1)} + \frac{c_{ijk}}{(d_i-1)(d_j-1)(d_k-1)})$, where c_{ij} (or c_{ik}, c_{jk}) is the number of common neighbors of i and j (or i and k, j and k), not including i, j, k; and c_{ijk} is the number of common neighbors of i, j and k. The increment is less than $4 \cdot \frac{1}{d_id_jd_k}$.

Therefore, the total increment over all the triad is less than $\delta = \sum_{\text{Type II}} \frac{1}{d_id_jd_k} + \frac{9}{4} \sum_{\text{Type III}} \frac{1}{d_id_jd_k} + 4 \sum_{\text{Type IV}} \frac{1}{d_id_jd_k}$ and after normalizing it by $—\Delta_H—$ we get: $\mathbb{E}_H(\frac{1}{d_id_jd_k}) < \mathbb{E}_G(\frac{1}{d_id_jd_k}) + \frac{\delta}{(n-3)\Delta_G}$.

Next, we compute $m_{3,H}$:

$$\begin{aligned}
m_{3,H} &= 2\,\mathbb{E}_H(\Delta_i)\,\mathbb{E}_H(\frac{1}{d_i d_j d_k}) \\
&< 2\frac{n-3}{n-1}\,\mathbb{E}_G(\Delta_i)(\mathbb{E}_G(\frac{1}{d_i d_j d_k}) + \frac{\delta}{(n-3)\Delta_G}) \\
&= 2(\frac{n-3}{n-1}\,\mathbb{E}_G(\Delta_i)\,\mathbb{E}_G(\frac{1}{d_i d_j d_k}) + \frac{\mathbb{E}_G(\Delta_i)\delta}{(n-1)\Delta_G}) \\
&= 2(\mathbb{E}_G(\Delta_i)\,\mathbb{E}_G(\frac{1}{d_i d_j d_k}) + \frac{\mathbb{E}_G(\Delta_i)\delta}{(n-1)\Delta_G} - \frac{2}{n-1}\,\mathbb{E}_G(\Delta_i)\,\mathbb{E}_G(\frac{1}{d_i d_j d_k})) \\
&= m_{3,G} + \frac{2\,\mathbb{E}_G(\Delta_i)}{(n-1)\Delta_G}(\delta - 2\Delta_G\,\mathbb{E}_G(\frac{1}{d_i d_j d_k})) \\
&= m_{3,G} + \frac{6}{n(n-1)}(\delta - 2\Delta_G\,\mathbb{E}_G(\frac{1}{d_i d_j d_k})) \quad (\text{as } E_G(\Delta_i) = \frac{3\Delta_G}{n}).
\end{aligned}$$

Notice that $\Delta_G\,\mathbb{E}_G(\frac{1}{d_i d_j d_k}) = \sum_{(i,j,k)\in G} \frac{1}{d_i d_j d_k}$, so $\delta - 2\Delta_G\,\mathbb{E}_G(\frac{1}{d_i d_j d_k})$
$= \frac{1}{4}\sum_{\text{Type III}} \frac{1}{d_i d_j d_k} + 2\sum_{\text{Type IV}} \frac{1}{d_i d_j d_k} - \sum_{\text{Type II}} \frac{1}{d_i d_j d_k} - 2\sum_{\text{Type I}} \frac{1}{d_i d_j d_k}$. The theorem is proved. □

5.6.3 Proof of Theorem 5.3.6

Proof. Similarly, there are n subgraphs $G_1, G_2, \ldots, G_n$. We construct $H = \bigcup_{i=1}^{n} G_i$, so $m_{4,H} = \frac{\sum_i m_{4,G_i}}{n} = \mathbb{E}(m_{4,G'})$. From Theorem 5.3.2, $m_4 = [\,\mathbb{E}(d_i) + 4\,\mathbb{E}\binom{d_i}{2} + 2\,\mathbb{E}(\square_i)\,]\,\mathbb{E}(\frac{1}{d_i d_j d_k d_l})$. In the formula, $\mathbb{E}(d_i)$ is the average degree; $\mathbb{E}\binom{d_i}{2}$ is the average number of wedges a node is in, so it equals to $\frac{w}{n}$ where w is the number of wedges; $\mathbb{E}(\square_i)$ is the average number of squares a node is in. $\mathbb{E}(\frac{1}{d_i d_j d_k d_l})$ is the expected value of $\frac{1}{d_i d_j d_k d_l}$ over all the closed walks of length 4. Hence, for graph H, we have $m_{4,H} = [\,\mathbb{E}_H(d_i) + 4\,\mathbb{E}_H\binom{d_i}{2} + 2\,\mathbb{E}_H(\square_i)\,]\,\mathbb{E}_H(\frac{1}{d_i d_j d_k d_l})$.

From Equation 5.1, we have $\mathbb{E}_H(d_i) = \frac{n-2}{n-1} \cdot \mathbb{E}_G(d_i)$; As each wedge in G has $n-3$ copies in H (when none of the three nodes is removed), $\mathbb{E}_H\binom{d_i}{2} = \frac{w_H}{n(n-1)} = \frac{(n-3)w_G}{n(n-1)} = \frac{n-3}{n-1} \cdot \mathbb{E}_G\binom{d_i}{2}$; Similarly, $\mathbb{E}_H(\square_i) = \frac{n-4}{n-1} \cdot \mathbb{E}_G(\square_i)$ as each square in G has $n-4$ copies in H.

To get $\mathbb{E}_H(\frac{1}{d_i d_j d_k d_l})$, of course we can discuss all the possible closed walks of length 4 (generated by edges, wedges or squares), as we have done in the previous theorems.

However, for a simpler exposition, we provide the following upper bound. Notice that $\mathbb{E}_H(\frac{1}{d_i d_j d_k d_l}) \leq \mathbb{E}_G(\frac{1}{(d_i-1)(d_j-1)(d_k-1)(d_l-1)})$ and the bound is tight when G is a complete graph ($n \geq 3$) or all of its components are k-cliques ($k \geq 3$), and $\frac{1}{(d_i-1)(d_j-1)(d_k-1)(d_l-1)} = \frac{1}{d_i d_j d_k d_l} \cdot \prod_{x\in\{i,j,k,l\}} \frac{d_x}{d_x-1} \leq \frac{16}{d_i d_j d_k d_l}$, so $\mathbb{E}_H(\frac{1}{d_i d_j d_k d_l}) \leq 16 \cdot \mathbb{E}_G(\frac{1}{d_i d_j d_k d_l})$. Therefore,

$$
\begin{aligned}
m_{4,H} &= [\,\mathbb{E}_H(d_i) + 4\,\mathbb{E}_H\binom{d_i}{2} + 2\,\mathbb{E}_H(\square_i)\,]\,\mathbb{E}_H(\frac{1}{d_i d_j d_k d_l}) \\
&\leq [\,\frac{n-2}{n-1}\,\mathbb{E}_G(d_i) + 4\frac{n-3}{n-1}\,\mathbb{E}_G\binom{d_i}{2} + 2\frac{n-4}{n-1}\,\mathbb{E}_G(\square_i)\,] \\
&\times 16 \cdot \mathbb{E}_G(\frac{1}{d_i d_j d_k d_l}) \\
&\leq \frac{n-2}{n-1}[\,\mathbb{E}_G(d_i) + 4\,\mathbb{E}_G\binom{d_i}{2} + 2\,\mathbb{E}_G(\square_i)\,] \times 16 \cdot \mathbb{E}_G(\frac{1}{d_i d_j d_k d_l}) \\
&= \frac{16(n-2)}{n-1} m_{4,G}
\end{aligned}
$$

□

5.7 Additional Related Work

I. Subgraph Spectrum. Past studies on the spectrum of subgraphs often focus on the Cauchy's interlacing theorem which bounds subgraph eigenvalues with the eigenvalues of whole graphs [106, 107]. However, interlacing theorem can only provide loose bounds on the spectral moments, whereas our bounds are either exact or tight.

II. Spectral Embedding. Recently, spectral information is used for different network embedding methods, such as FGSD [98] and NetLSD [30]. Compared to them, spectral path is easy to be interpreted and utilizes the spectral information of subgraphs.

5.8 Conclusion

We propose representing a network with a 3D spectral path: a path connecting the spectral moments of a network to the expected spectral moments of its subgraphs. We demonstrate the interpretability of spectral paths. We show the utility of spectral paths in network visualization, network identification and distinguishing cospectral graphs. To the best of our knowledge, this is the first study to explore spectral moments of subgraphs and study the relationship between spectral moments of subgraphs and those of the whole network.

Limitations and Future Work. For a graph of n nodes, there are $\binom{n}{k}$ subgraphs of size k. When k is around $\frac{n}{2}$, the number of subgraphs is the largest. Hence, one may consider taking different numbers of samples to estimate the expected moments. In our experiments on real-world networks, we take 20 samples for each sampling proportion, and the standard deviation of spectral moments is often less than 5%. In the future, we aim to study the distribution of spectral moments of subgraphs.

Chapter 6

Spectral Moments and Network Robustness

6.1 Introduction

In Chapter 4 and 5, we propose embedding a graph with its spectral points, and we introduce spectral path, which represent a network with a 3D path in the spectral embedding space. We have shown that both representations provides interpretability, especially on network structural properties of a network and its subgraphs. In this chapter, we will discuss the connection between spectral moments and network robustness.

The study of network robustness in complex systems plays an important role in various fields such as biology, economics, and engineering. Network robustness is often defined as *a network's ability to continue functioning when part of the network is either naturally damaged or targeted for attack* [108–110]. In the study of network robustness, there are two fundamental research goals: (1) the assessment of the robustness of a network, i.e., how to quantify the network robustness? (2) the utility of network robustness, i.e., how to use the robustness of a network? In this chapter, we aim to have a systematic study on network robustness, by answering the following three questions: [**Q1**] how to assess the robustness of networks?; [**Q2**] how to design networks with controlled robustness?; [**Q3**] how to study the behavior of a complex system by observing the evolution of its network robustness?

The Present Work: Spectral Moments for Network Robustness Assessment. In this

chapter, we propose utilizing spectral moments, especially the second spectral moment m_2 of the random walk transition matrix of a network as a robustness measure, justified by various reasons: (a) **Capture network robustness**. We prove that spectral moments are tightly connected to existing network robustness measures including average distance, diameter, spectral radius, and the existence of a giant component; (b) **Interpretablity**. Spectral moments have been used to capture the shape of a spectral density, and they have been proved to capture various network structures and properties in Section 4 and 5. Specifically, m_2 has a clear meaning, which is the expected return probability of a 2-step random walk. Intuitively, in a graph with a small expected return probability for a random walk (a walk which travels far away from its starting node) is more likely an indication of a well-connected graph. This observation motivates the use of m_2 as a measure of network robustness. (c) **Easy and fast to compute.** For large networks, m_2 can be approximated accurately in seconds.

Overall, our contributions are mainly the following:

I. A Spectral Measure for Network Robustness. We propose using the second spectral moment m_2 of a network as a network robustness measure. We show that m_2 can capture network robustness on both synthetic and real-world networks. Specifically, when m_2 is smaller, the network is more robust. The spectral moments can be used to assess the degree of robustness of a network, or to compare the robustness of two networks varying in size.

II. Connection to Existing Network Robustness Measures. We prove that the second spectral moment m_2 is closely related to four well-known robustness measures (average distance, diameter, spectral radius, and the existence of a giant component) for random graphs with given expected (or exact) degree sequences.

III. Designing Networks with Controllable Robustness. We show that we can control the network robustness by manipulating its m_2 value, to design a network that is more robust under failures. We conduct experiments and evaluation on real-world networks.

IV. Evolution of Network Robustness under Cascading Failures. We demonstrate that with m_2 as the robustness measure, one can study how a complex networked system behaves under cascading failures by looking at how network robustness evolves. By studying cascading failures in a power grid network, we show that after an initial failure making the grid vulnerable, the grid stabilizes after the cascading failures.

The rest of the chapter is organized as follows. We propose the use of m_2 as a network robustness measure, and we show the relationship between the second spectral moment and other robustness measures in Section 6.2. We use the second spectral moment to assess robustness of real-world networks in Section 6.3, and discuss ways to design networks with controllable robustness in Section 6.4. Section 6.5 details our observations on the evolution of robustness under cascading failures in a power grid. After reviewing further related work in Section 6.6, we conclude in Section A.5.

6.2 Spectral Moments as a Robustness Measure

As we have mentioned, we propose using the second spectral moments m_2 of random walk transition matrix as a network robustness measure.

6.2.1 Second spectral moment m_2 and the Estrada Index

As mentioned above, m_2 is the expected return probability of a 2-step random walk. Naturally, one may have a valid concern that it does not directly capture robustness in terms of higher-order information, i.e., the return probability of longer walks. Here, we show that, on the contrary, m_2 actually provides tight upper and lower bounds on the expected return probability of a random walk of any length, discounting longer walks. For that, we first introduce the normalized Estrada index of the random-walk transition matrix $EE_{P-norm}(G)$.

The Estrada index of a graph G is defined as $EE(G) = \sum_{j=1}^{n} e^{\mu_j}$, where μ_j's are the

eigenvalues of the adjacency matrix A [83]. The Estrada index counts the number of closed walks, discounting longer walks, as $EE(G) = \text{trace}(e^A) = \sum_{k=0}^{\infty} \frac{\text{trace}(A^k)}{k!}$. Therefore, Estrada index is sometimes used to measure the global connectivity of a graph. In [15], a variation of the Estrada index using the random walk transition matrix P is denoted as $EE_P(G) = \sum_{j=1}^{n} e^{\lambda_j} = \sum_{j=1}^{n} \text{trace}(e^P) = \sum_{k=0}^{\infty} \frac{\text{trace}(P^k)}{k!} = n \sum_{k=0}^{\infty} \frac{m_k}{k!}$. Unlike the Estrada index, $EE_P(G)$ computes the expected return probability of a random walk of any length, discounting longer walks. Intuitively, if a walk can travel far away from its starting node, it is an indication that the graph is well-connected. Generally, the smaller the $EE_P(G)$ value, the more well-connected the graph G. Here, we normalize $EE_P(G)$ by the size of the graph and get $EE_{P-norm}(G) = \frac{1}{n} EE_P(G) = \sum_{k=0}^{\infty} \frac{m_k}{k!}$, to cancel the effect of the size of the graph.

In Theorem 6.2.1, we prove that the second moment m_2 provides both tight upper and lower bounds on $EE_{P-norm}(G)$. In other words, the expected return probability of longer random walks can be bounded by functions of m_2.

Theorem 6.2.1 (Bounds on $EE_{P-norm}(G)$ by m_2)**.** *For an undirected graph G without self-loops, its normalized Estrada Index $EE_{P-norm}(G)$ is bounded by the second moment m_2:*

$$1 + \frac{m_2}{2} \leq EE_{P-norm}(G) \leq 1 + m_2$$

Proof. We first prove $1 + \frac{m_2}{2} \leq EE_{P-norm}(G)$. For an undirected graph without self-loops, it is clear that $m_0 = 1$ and $m_1 = 0$. By definition, $EE_{P-norm}(G) = \sum_{k=0}^{\infty} \frac{m_k}{k!} \geq \sum_{k=0}^{2} \frac{m_k}{k!} = m_0 + m_1 + \frac{m_2}{2} = 1 + \frac{m_2}{2}$, as $m_k \geq 0$.

Next, we prove $EE_{P-norm}(G) \leq 1 + m_2$. As $e^x \leq 1 + x + x^2$, $EE_{P-norm}(G) = \frac{1}{n} EE_P(G) = \frac{1}{n} \sum_{j=1}^{n} e^{\lambda_j} \leq \frac{1}{n} \sum_{j=1}^{n} (1 + \lambda_j + \lambda_j^2) = 1 + m_1 + m_2 = 1 + m_2$.

The bounds are tight; consider an empty graph. Then, $m_k = 0$ for $k \geq 1$. Hence, $m_2 = 0$ and $EE_{P-norm}(G) = 1$. □

6.2.2 Spectral Moments and Existing Robustness Measures

Next, we connect m_2 with four well-known robustness measures: diameter, average distance, spectral radius, and giant component. Particularly, we show that spectral moments are connected to the robustness of graphs generated by two network models: Chung-Lu and Configuration Model.

Consider a random graph with an expected degree sequence (also known as the *Chung-Lu model* [111, 112]). Chung-Lu model is a general model $G(\mathbf{w})$ for random graphs with a given expected degree sequence $\mathbf{w} = (w_1, w_2, \ldots, w_n)$. For a random graph $G \in G(\mathbf{w})$, the edge between nodes v_i and v_j is chosen independently with probability $p_{ij} = \frac{w_i w_j}{\sum_i w_i}$, which is proportional to the product $w_i w_j$. Denote $\tilde{d} = \frac{\sum w_i^2}{\sum w_i}$ as the second-order average degree. Chung et al. have shown that $\tilde{d}$ is closely related to various graph properties [111–113]. In the rest of the chapter, a random graph G with degree sequence $(d_1, d_2, \ldots, d_n)$ refers to one realization of those generated by the Chung-Lu model, i.e., $G \in G(\mathbf{w})$ where $\mathbf{w} = (d_1, d_2, \ldots, d_n)$. We show that the spectral moments of the graphs generated by the Chung-Lu model capture various robustness measures in them.

Similarly, we consider random graphs generated by the configuration model (Molloy-Reed model), where the graph has a fixed degree sequence. We show that spectral moments also capture robustness, in terms of the existence of the giant component, in such graphs.

We start with the Chung-Lu model and in Lemma 6.2.2, we demonstrate that second-order average degree $\tilde{d}$ is lower bounded by the inverse of second spectral moment m_2 of a Chung-Lu random graph.

Lemma 6.2.2. *For a random graph G with given expected degrees, the second-order average degree $\tilde{d}$ satisfies*

$$\tilde{d} \geq \sqrt{\frac{\mathbb{E}(d_i)}{m_2}},$$

where m_2 is the second spectral moment of G and $\mathbb{E}(d_i)$ is the average node degree in G.

Proof. By definition, $\tilde{d} = \frac{\sum w_i^2}{\sum w_i} = \frac{\sum d_i^2}{\sum d_i} = \frac{\mathbb{E}(d_i^2)}{\mathbb{E}(d_i)}$. From Theorem 4.1 of [15], for any graph $\mathbb{E}(\frac{1}{d_i d_j}) \geq \frac{\mathbb{E}^2(d_i)}{\mathbb{E}^2(d_i^2)}$, so $\mathbb{E}(\frac{1}{d_i d_j}) \geq \frac{1}{\tilde{d}^2}$, implying $\tilde{d}^2 \geq \frac{1}{\mathbb{E}(\frac{1}{d_i d_j})}$ and $\tilde{d} \geq \sqrt{\frac{1}{\mathbb{E}(\frac{1}{d_i d_j})}}$. By Thm. 4.3.2, $m_2 = \mathbb{E}(d_i)\,\mathbb{E}(\frac{1}{d_i d_j})$, so $\tilde{d} \geq \sqrt{\frac{\mathbb{E}(d_i)}{m_2}}$. □

Next, we will show the connection between spectral moments with the following robustness measures: (1) average distance, (2) diameter, and (3) spectral radius of a graph with a given expected degree distribution.

I. Average Distance. In a graph G, denote distance $d(u, v)$ as the length of the shortest path between u and v. Average distance of a graph G, denoted by $\overline{d_{uv}}$, is the average distance over all pairs of vertices (u, v) in G. A smaller $\overline{d_{uv}}$ shows that nodes are closer to each other and the network is well-connected and more robust [109].

Theorem 6.2.3. *For a random graph G with given expected degree sequence, if $\mathbf{w} = (d_1, d_2, \ldots, d_n)$ is admissible, for the average distance $\overline{d_{uv}}$, we have*

$$\overline{d_{uv}} \leq (1 + o(1))\frac{2\log n}{\log \mathbb{E}(d_i) - \log m_2}.$$

Proof. From [113], the average distance $\overline{d_{uv}}$ is almost surely $(1 + o(1))\frac{\log n}{\log \tilde{d}}$, when the degree sequence is *admissible* (see definition in [113]). Specifically, $\overline{d_{uv}}$ is upper bounded by $(1 + o(1))\frac{\log n}{\log \tilde{d}}$. By Lemma 6.2.2, $\tilde{d} \geq \sqrt{\frac{\mathbb{E}(d_i)}{m_2}}$. Moreover, in our settings, d_i's are the degree of the nodes, so $d_i \geq 1$ or $d_i = 0$, and $\tilde{d} = \frac{\sum d_i^2}{\sum d_i} \geq 1$, and $\frac{\mathbb{E}(d_i)}{m_2} = \frac{1}{\mathbb{E}(\frac{1}{d_i d_j})} \geq 1$. Therefore, $\frac{1}{\log \tilde{d}} \leq \frac{1}{\log \sqrt{\frac{\mathbb{E}(d_i)}{m_2}}}$. Hence, $\overline{d_{uv}} \leq (1 + o(1))\frac{\log n}{\log \sqrt{\frac{\mathbb{E}(d_i)}{m_2}}} = (1 + o(1))\frac{2\log n}{\log \mathbb{E}(d_i) - \log m_2}$. □

From Theorem 6.2.3, for a random graph with a given expected degree distribution (naturally, n and $\mathbb{E}(d_i)$ is fixed), the average distance of the graph is upper bounded by a term that depends on m_2. Specifically, when m_2 is smaller, the upper bound is smaller. Hence, in terms of the average distance, a smaller m_2 indicates a more robust network.

II. Diameter. The diameter of graph G, denoted by $D(G)$, is the maximum distance over all pairs of nodes in G. The diameter is closely connected to robustness, as it is a tight upper bound on the distance between any two nodes in the network. Thus, a smaller diameter shows more robustness [109].

Theorem 6.2.4. *For a random graph G with given expected degree sequence, if $\mathbf{w} = (d_1, d_2, \ldots, d_n)$ is specially admissible, the diameter $D(G)$ is almost surely $\mathcal{O}(\frac{2\log n}{\log \mathbb{E}(d_i) - \log m_2})$.*

Proof. From [113], $D(G)$ is almost surely $\Theta(\frac{\log n}{\log \tilde{d}})$, when the degree sequence is *specially admissible* (see definition in [113]). As $\frac{1}{\log \tilde{d}}$ is upper bounded by $\frac{1}{\log \sqrt{\frac{\mathbb{E}(d_i)}{m_2}}}$, we have $D(G)$ is almost surely $\mathcal{O}(\frac{2\log n}{\log \mathbb{E}(d_i) - \log m_2})$. □

Similar to the average distance, Theorem 6.2.4 shows that the diameter of a random graph G with a given degree sequence is upper bounded by a term that depends on m_2. Specifically, when m_2 is smaller, the upper bound on the diameter is smaller. Hence, in terms of the diameter, a smaller m_2 indicates a more robust network.

III. Spectral Radius. The largest eigenvalue of the adjacency matrix A is called its spectral radius ρ. The spectral radius is closely related to the *path capacity* or *loop capacity* of the graph. A larger ρ implies that the graph has many loops and paths, so the graph is well-connected [114, 115]. In general, a larger ρ indicates a more robust network.

Theorem 6.2.5. *For a random graph with given expected degree sequence, if $\tilde{d} > \sqrt{d_{\max}(G)} \log n$, then $\rho \geq (1 + o(1))\sqrt{\frac{\mathbb{E}(d_i)}{m_2}}$, where $d_{\max}(G)$ is the maximum degree.*

Proof. Chung et. al [112] proved that when $\tilde{d} > \sqrt{d_{\max}(G)} \log n$, ρ is roughly equal to the the second order average degree $\tilde{d}$, i.e., ρ is almost surely $(1 + o(1))\tilde{d}$, and especially ρ is lower bounded by $(1 + o(1))\tilde{d}$ [112, 116]. By Lemma 6.2.2, we get $\rho \geq (1 + o(1))\sqrt{\frac{\mathbb{E}(d_i)}{m_2}}$.

□

Theorem 6.2.5 indicates that if m_2 is smaller, then ρ has a greater lower bound. Hence, in terms of the spectral radius, a smaller m_2 indicates a more robust network.

Finally, we show that even when in the random graph the degree sequence is fixed, the spectral moments are related to network robustness. For that, we consider the graphs generated by the configuration model (Molloy-Reed model) and show that spectral moments capture the existence of the giant component.

IV. Giant Component. For a graph $G = (V, E)$, a giant component of G is a connected component having at least $\mathcal{O}(|V|)$ nodes [117,118]. A component is called c-giant if it has at least $c \cdot |V|$ nodes (or $c \cdot |E|$ edges) [111]. In studies of network robustness, c is often defined as the fraction of nodes contained in the largest connected component, to measure network availability i.e., what percentage of the nodes can be reached [109]. Though the existence of a giant component does not mean that the network is robust (as in some cases the component can be split into small components by losing a few edges due to *bridges* in the network), it shows that the network keeps most nodes and maintains "functionality." In Theorem 6.2.6, we show that m_2 can capture the existence of the giant component for Molloy-Reed random graphs.

Theorem 6.2.6. *For a random graph G with an exact degree sequence generated by the Molloy-Reed model, when $m_2 < \frac{1}{4}\,\mathbb{E}(d_i)$, a giant component exists.*

Proof. Molloy-Reed Criterion states that for a random graph G generated by the Molloy-Reed model, when $\kappa = \frac{\mathbb{E}(d_i^2)}{\mathbb{E}(d_i)} > 2$, a giant component exists [119, 120]. Similar to Lemma 6.2.2, we can show that $\kappa \geq \sqrt{\frac{1}{\mathbb{E}(\frac{1}{d_i d_j})}}$. Thus, if $\mathbb{E}(\frac{1}{d_i d_j}) < \frac{1}{4}$, we can ensure $\kappa > 2$ and a giant component exists. Further, the condition $\mathbb{E}(\frac{1}{d_i d_j}) < \frac{1}{4}$ is equivalent to $m_2 < \frac{1}{4}\,\mathbb{E}(d_i)$, proving the theorem. □

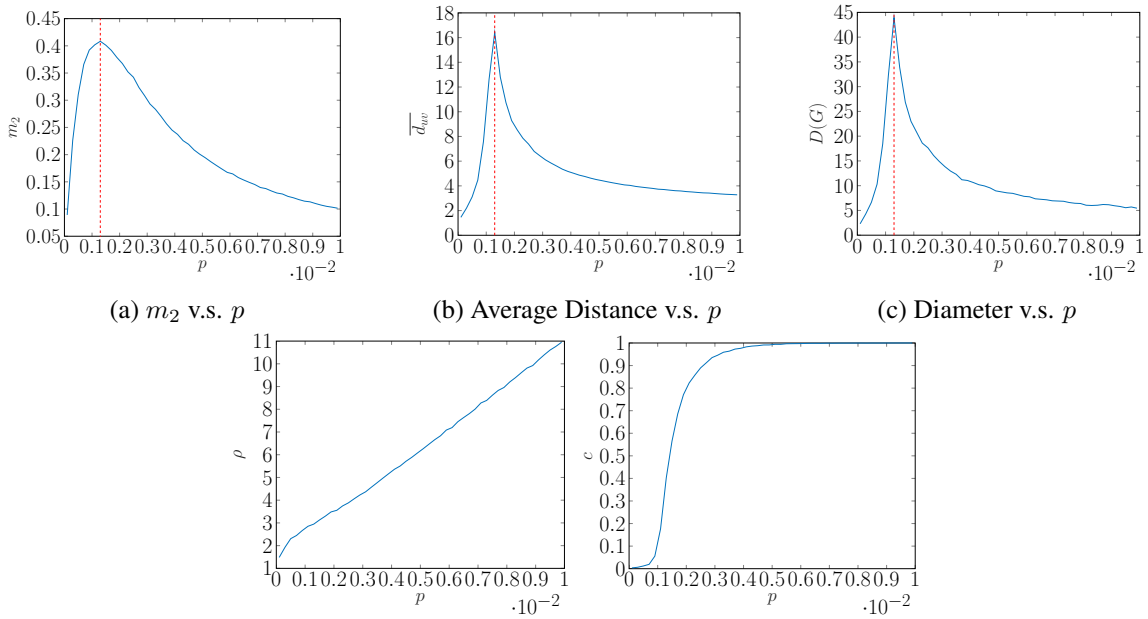
For a random graph G with an exact degree sequence, the average degree $\mathbb{E}(d_i)$ is fixed. Hence, from Theorem 6.2.6, we find that for such a graph when m_2 is smaller, it is more

likely to have a giant component.

6.2.3 Experiments on Synthetic Networks

We have shown the theoretical connection between m_2 and existing robustness measures. Here, we explore this connection empirically as well. To that end, we generate synthetic networks using the random graph model $G(n, p)$. For random graphs generated by $G(n, p)$, the behaviour of the size of the largest component is well-studied for p near $\frac{1}{n}$. For $p < \frac{1}{n}$, the size of the largest component is almost surely $O(\log n)$; for $p = \frac{1}{n}$, the size of the largest component is almost surely $\Theta(n^{2/3})$; and for $p > \frac{1}{n}$ the size of the largest component is almost surely $\Theta(n)$ [117, 118, 121]. For $p > \frac{1}{n}$, this largest component is commonly referred to as the *giant component* of $G(n, p)$, and the point $p = \frac{1}{n}$ is referred to as the *critical point* (for the *phase transition*). Here, we study the behavior of the second spectral moment m_2 and other network robustness measures near this critical point.

In our experiments, we set $n = 1,000$ nodes and vary p from 0.0001 to 0.01 with step size 0.0002. For each variation, we generate 20 random graphs, and in Figure 6.1, we plot the average value of m_2, $\overline{d_{uv}}$ (the average distance), $D(G)$ (the diameter), ρ (the spectral radius) and c (the fraction of nodes in the largest connected component). When the graph is not connected, we use $\overline{d_{uv}}$ and $D(G)$ of its largest connected component. We find that (1) with the increase of p, m_2 has a similar changing pattern to $\overline{d_{uv}}$ and $D(G)$: they all increase first and then decrease; (2) all of the turning points are at $p = 0.0013$, which is slightly greater than the critical point $p = 0.001$. In essence, the average distance and diameter increase with p when there is no giant component in the graph. However, when the giant component emerges, they keep increasing until a certain point and start to decrease. Our results show that m_2 captures this behavior well. Note that the time complexity to compute the average distance and diameter both requires $O(n^3/2^{\Omega(\log n)^{1/2}})$ [122] which is not feasible for large networks, but m_2 can be computed in a few seconds. Next, we look



(a) m_2 v.s. p (b) Average Distance v.s. p (c) Diameter v.s. p

(d) Spectral Radius ρ v.s. p (e) Fraction of Nodes in Largest Connected Component c v.s. p

Figure 6.1: $n = 1,000$ and dashed line shows the turning point at $p = 0.0013$.

into using the second spectral moment m_2 to assess robustness in real-world networks.

6.3 Assess Robustness in Real-World Networks

In this section, we aim to investigate spectral moment m_2 as a network robustness measure in real-world networks in Chapter 4, and Table 6.1 lists the m_2 value of each network. We aim to answer the question: [**Q1**] how to assess the robustness of networks with m_2? Therefore, we need to understand the connection between the robustness of a real-world network and its second spectral moment m_2. In other words, should a robust network have a larger or smaller m_2 value? Before presenting experiments, we review experimental setup.

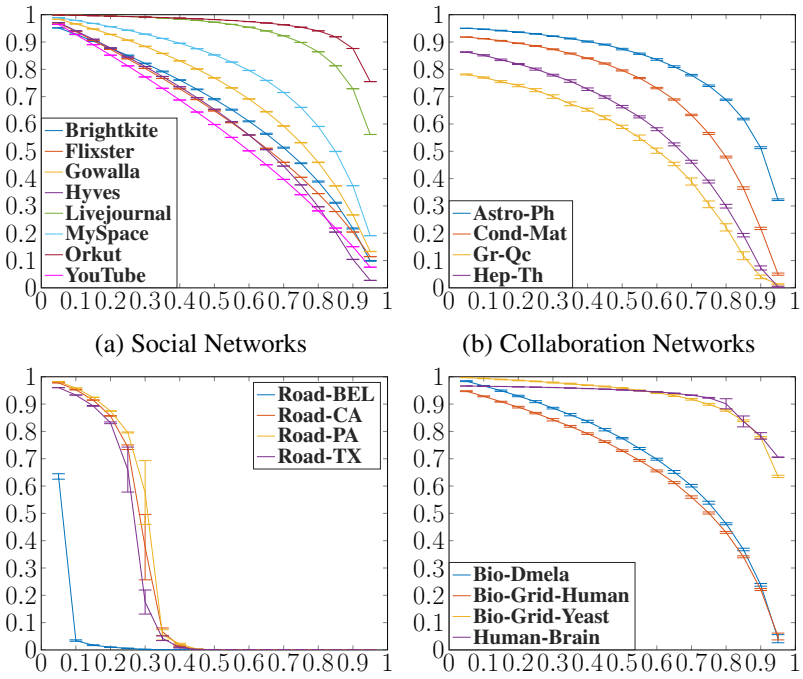
6.3.1 Assess Network Robustness with Spectral Moments

To evaluate m_2 as a network robustness measure, we first define robustness of a real-world network. In its most abstract form, *robustness is the ability of a network to continue to*

Table 6.1: Dataset Statistics

Type	Network	m_2
Social Networks	Brightkite	0.1799
	Flixster	0.0261
	Gowalla	0.1403
	Hyves	0.0610
	Livejournal	0.0174
	MySpace	0.0923
	Orkut	0.0187
	YouTube	0.1574
Collaboration Networks	Astro-Ph	0.1007
	Cond-Mat	0.1672
	Gr-Qc	0.2831
	Hep-Th	0.2488
Road Networks	Road-BEL	0.4646
	Road-CA	0.3545
	Road-PA	0.3557
	Road-TX	0.3577
Biological Networks	Bio-Dmela	0.1278
	Bio-Grid-Human	0.1787
	Bio-Grid-Yeast	0.0198
	Human-Brain	0.0236

perform well under failures or attacks [108]. To quantify such a definition in our experiments, we consider the robustness of a network by looking at how c – the fraction of nodes in its largest connected component – changes under random edge failures. In other words, when losing the same number (or proportion) of edges, a more robust network exhibits a smaller drop in c value as most nodes within the "core" of the network are kept intact. Hence, for each network, we randomly remove $x\%$ of the edges of the graph by varying $x\%$ from 5% to 95% with step size 5%. For each $x\%$, we run the experiments 20 times and report the average c and its standard deviation in Figure 6.2. From the figure, we find that (1) road networks are much more vulnerable under random failures. For each road network, the size of its largest component drops sharply when losing edges randomly. Especially, by losing 35% of the edges, c becomes less than 10%. We notice that m_2 values of road networks are much larger than those of networks from other categories. Among road networks, Road-BEL is more vulnerable than others and has the largest m_2; (2) for networks from other three categories, c decreases smoothly as more edges are removed.



(a) Social Networks (b) Collaboration Networks

(c) Road Networks (d) Biological Networks

Figure 6.2: x value: fraction of edges removed; y value: fraction of nodes in the largest connected component c.

Furthermore, if a network has a larger m_2, the fraction of nodes in its largest component shrinks faster. For networks with smaller m_2 values, such as Orkut and Human-Brain, they maintain more than 70% of the nodes in their largest component even after losing 90% of their edges. In general, these observations provide an answer to **Q1**: a real-world network with a smaller second spectral moment m_2 is more robust under random failures. Hence, we can compare the robustness of two networks by comparing their m_2 values, even if the networks vary in size.

6.4 Design Networks with Controllable Robustness

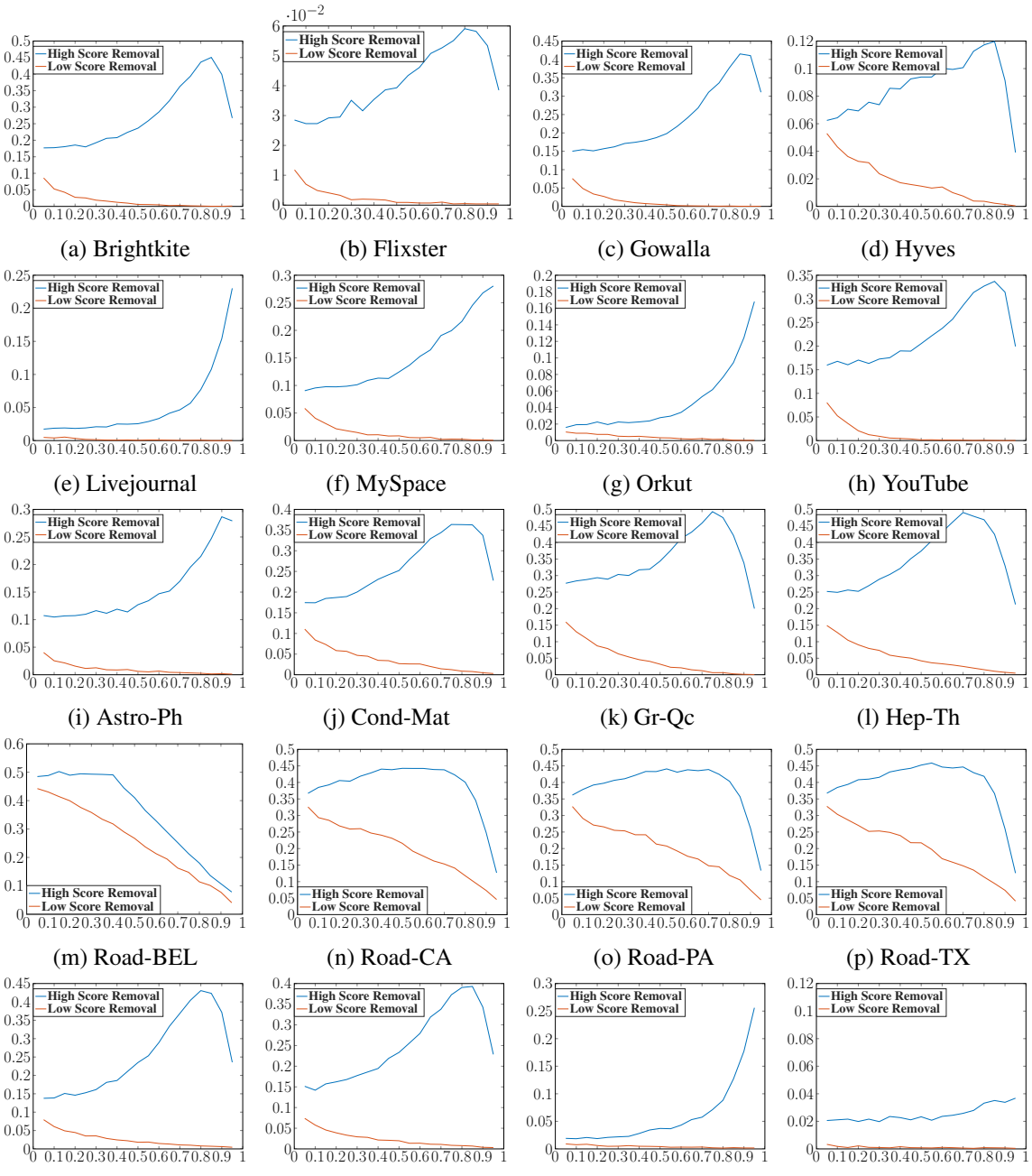
Next, we want to answer the question: [**Q2**] how to design networks with controlled robustness? In other words, can we design strategies to control (increase or decrease) robustness in a real-world network? From Section 6.3, we know that a robust network has a smaller m_2 value. Naturally, if we can control the network robustness by manipulating its m_2 value, we can "design" a network that is more robust under failures; or equivalently,

develop more efficient attack models to harm the robustness of a network. Thus, we will design various edge removal strategies here and assess their impact on the m_2 value of a network.

Theorem 4.3.2 shows that $m_2 = \mathbb{E}(d_i)\,\mathbb{E}(\frac{1}{d_i d_j})$. Assume we remove a fixed number of edges from some graph G to get a new graph G'. The average degree of G' will only rely on the number of edges removed and is independent of which edges were removed from graph G. Hence, when a fixed number of edges are removed, what can make m_2 different is how these removed edges change the value of $\mathbb{E}(\frac{1}{d_i d_j})$. Intuitively, by removing edges (i, j) corresponding to higher $d_i d_j$ values (d_i and d_j are the degress of i and j), we should get a larger value of $\mathbb{E}(\frac{1}{d_i d_j})$ in G'. Hence, we design edge removal strategies that rely on the $d_i d_j$ values of edges. Here, we detail the developed edge removal strategies.

We define $d_i d_j$ value as the edge score for an edge (i, j) between nodes i and j with degrees d_i and d_j. We propose two strategies to remove edges based on the edge score: (1) *High Score Removal*, removing the edges with the highest scores from the graph; and (2) *Low Score Removal*, which removes the edges with the lowest scores. When an edge is removed from the graph, the scores of edges incident to the endpoints of the removed edge will change, which may impact the current ranking of edges based on this edge score. Hence, for the removal process, we propose two methods: (1) *Batch Removal*, where we pick top $x\%$ of edges in the graph based on each strategy (high score or low score removal) and remove them in one batch; (2) *Sequential Removal*, where each time we remove only the top-1 edge based on each strategy and after each removal, we update the ranks. In total, we remove $x\%$ of edges of the graph. For both methods, we vary $x\%$ from 5% to 95% with the step size 5%, and we report the changes in m_2 for networks in Figure 6.3 and 6.4.

We observe that for both batch and sequential removal: (1) for *High Score Removal*, with more edges removed, m_2 of most networks increases first and after a certain point, m_2



(a) Brightkite (b) Flixster (c) Gowalla (d) Hyves

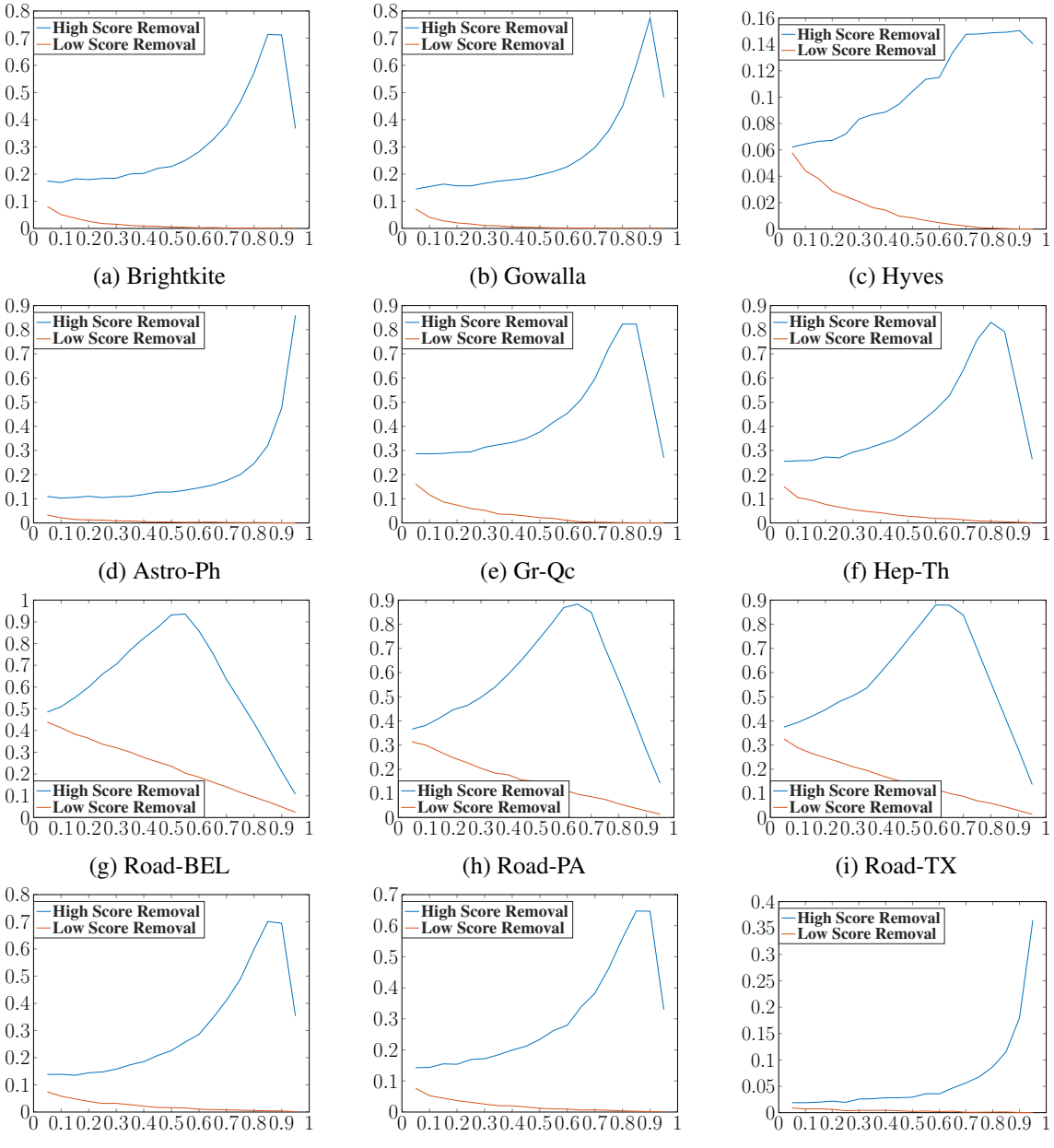
(e) Livejournal (f) MySpace (g) Orkut (h) YouTube

(i) Astro-Ph (j) Cond-Mat (k) Gr-Qc (l) Hep-Th

(m) Road-BEL (n) Road-CA (o) Road-PA (p) Road-TX

(q) Bio-Dmela (r) Bio-Grid-Human (s) Bio-Grid-Yeast (t) Human-Brain

Figure 6.3: x: proportion of edges removed; y: m_2.



(a) Brightkite (b) Gowalla (c) Hyves

(d) Astro-Ph (e) Gr-Qc (f) Hep-Th

(g) Road-BEL (h) Road-PA (i) Road-TX

(j) Bio-Dmela (k) Bio-Grid-Human (l) Bio-Grid-Yeast

Figure 6.4: x: fraction of removed edges; y: m_2. (Note: For large graphs with more than 5 million edges such as Livejournal and Orkut, sequential edge removal is too time-consuming to compute, as sequential edge removal requires update on the order each time we remove only the top-1 edge. Therefore, for each category we choose three networks.)

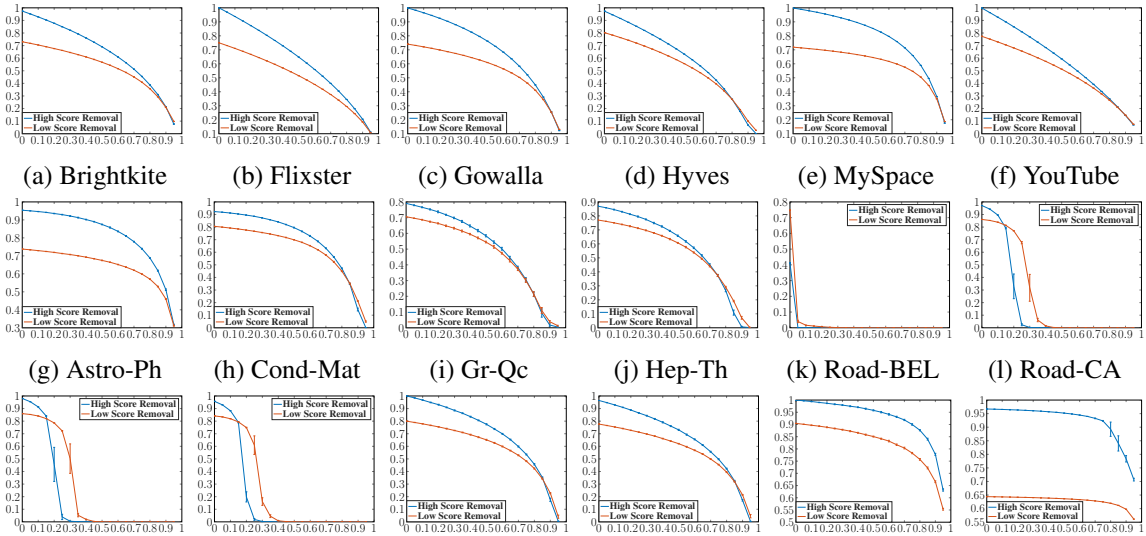
Table 6.2: Phase Transition of m_2

Network	Proportion of Edges Removed (Turning Point)	Average Degree of the Remaining Graph
Brightkite	0.85	7.353×0.15 = 1.10
Flixster	0.85	6.276×0.15 = 0.94
Gowalla	0.90	9.668×0.10 = 0.97
Hyves	0.85	3.960×0.15 = 0.59
YouTube	0.85	5.265×0.15 = 0.79
Astro-Ph	0.90	21.10×0.10 = 2.11
Cond-Mat	0.85	8.078×0.15 = 1.21
Gr-Qc	0.75	5.526×0.25 = 1.38
Hep-Th	0.70	5.259×0.30 = 1.66
Road-BEL	0.40	2.143×0.60 = 1.29
Road-CA	0.65	2.816×0.35 = 0.99
Road-PA	0.65	2.834×0.35 = 0.99
Road-TX	0.55	2.785×0.45 = 1.25
Bio-Dmela	0.85	6.917×0.15 = 1.04
Bio-Grid-Human	0.85	13.09 × 0.15 = 1.96

drops sharply. Further, if we look at the turning point of the curve, it always happens when the average degree of the remaining graph is around 1.0 (see Table 6.2), indicating a *phase transition* for m_2. However, if a network has a very high average degree (such as Bio-Grid-Yeast or Orkut), by removing 95% of its edges, the average degree of the remaining graph can be much greater than 1.0. For such networks, the phase transition will not appear in the figures; (2) for *Low Score Removal*, m_2 decreases monotonously as more edges are removed. So, generally, in response to **Q2**, removing edges (i, j) corresponding to highest $d_i d_j$ values decreases network robustness (increases m_2), and removing edges corresponding to lowest $d_i d_j$ values increases network robustness (decreases m_2).

6.4.1 Evaluation

We evaluate whether the proposed manipulations on m_2 can change network robustness. For a network G, we first remove 10% of its edges with *High Score Removal* (and *Low Score Removal*) in batch to get G_{High} (and G_{Low}); then we let G_{High} (and G_{Low}) experience the same random edge failures as detailed in Section 6.3.1. The results are shown in Figure 6.5. From the figure, we find that (1) we initially observe in G_{Low} a smaller largest



(a) Brightkite (b) Flixster (c) Gowalla (d) Hyves (e) MySpace (f) YouTube

(g) Astro-Ph (h) Cond-Mat (i) Gr-Qc (j) Hep-Th (k) Road-BEL (l) Road-CA

(m) Road-PA (n) Road-TX (o) Bio-Dmela (p) Bio-Grid-Human (q) Bio-Grid-Yeast (r) Human-Brain

Figure 6.5: (Note: Due to the large size, Livejournal and Orkut are not included in this experiment.)

connected component, as low degree nodes are removed from the component. However, this observation does not mean that G_{Low} is vulnerable as the remaining nodes in the component can be well-connected; (2) In terms of the robustness, G_{High} is more vulnerable under random failures. By looking at the slope of the curve, we observe that when under the same random failures (randomly losing the same number of edges), the size of the largest connected component of G_{High} shrinks faster than that of G_{Low}. Hence, *High Score Removal* increases m_2 of a network, making it less robust.

6.5 Evolution of Network Robustness under Cascading Failures

Next, we are going to answer the question: [**Q3**] how to study the behavior of a complex system by observing the evolution of its network robustness? We specifically consider the evolution of network robustness under cascading failures. In reality, in a network-based system the activity of an edge (or a node) often depends on the activity of its neighboring edges (or nodes) [123]. Hence, the failure of an edge can trigger the failure of the edges

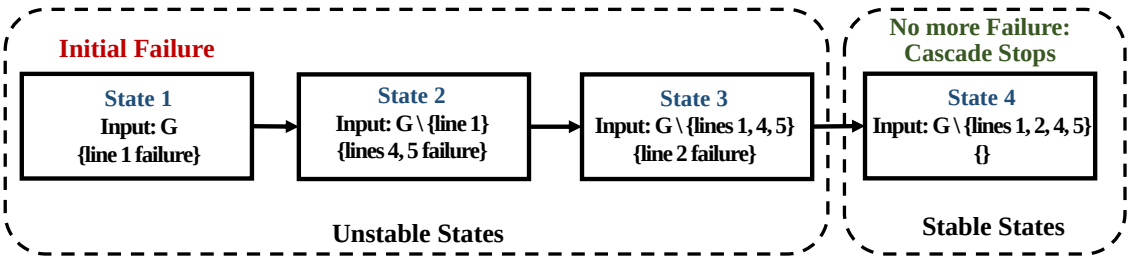


Figure 6.6: A Cascade Example

incident to it, and such sequences of failures are called *cascading failures*. For example, a power grid network is composed of busses (nodes) and transmission lines (edges). If one (or multiple) transmission lines are disconnected (e.g., due to natural disasters or operator mistakes), it can cause some other transmission lines to fail by exceeding their power flow limit and trigger more failures. Different from random failures or failures caused by attacks, cascading failures are closely related to the governing laws of the underlying networked system, e.g., power flow equations. Hence, during cascading failures, how a network evolves in terms of its robustness can indeed shed light on the governing laws of the underlying system.

6.5.1 Data Collection

We study the cascading failures in a well-studied power grid network (see [124] for details). We generate the cascading failures with the methods provided by Ma et al. [23]. We sample 100,000 different initial loading conditions on this power grid. For each initial loading condition, we choose all single-line failures as the initial failures. Then, we use the AC-based power flow to obtain the cascading failures. As this power grid has 41 transmission lines, we have $41 \times 100,000 = 4,100,000$ initial failure events in total. Among these initial failures, 1,644,135 of them trigger a cascading failure sequence. Figure 6.6 provides an example. In this example, we define state 4 as the *stable state* of the cascade, and other states as *unstable states* as they trigger subsequent failures of power lines.

m_2	Number of Patterns	Proportions
$\nearrow$	2,466,379	46.4%
$\searrow$	2,836,207	53.4%
$\rightarrow$	10,583	0.2%

Table 6.3: Changing Pattern of m_2 in Cascading Failures. $\nearrow$: m_2 increases; $\searrow$: m_2 decreases; $\rightarrow$: m_2 does not change.

6.5.2 Analysis

In a cascade, at each state, the system can be viewed as a subgraph of the previous states as we are losing power lines (edges). Thus, we can view each cascade as a sequence of subgraphs. We represent each cascade using the m_2 values of its subgraphs, and we study the changing patterns of m_2 between consecutive states. For example, if a cascade has four states and m_2 values of the sequence of subgraphs are: $[0.3632, 0.3893, 0.3726, 0.3514]$, then the changing patterns are $\nearrow\searrow\searrow$ which is composed of one increase and two decrease of m_2. Across all the cascades, we report the total number of changing patterns in Table 6.3. We find that in general, decreasing patterns (53.4%) are slightly more than the increasing patterns (46.4%). Next, for each cascade, we compare the m_2 of the initial failure state and that of the final (stable) state. Table 6.4 demonstrates that for 76.7% of the cascades, the m_2 value of the final state is smaller than that of the initial state, compared to 23.2% on the other direction. The difference is much more significant than that of the consecutive changing patterns. Notice that a smaller m_2 indicates the network is more robust. Hence, in general, an initial failure happens at a vulnerable state, and after the cascading failures change system robustness, the system stabilizes (converges to a more robust network).

m_2	Number of Cases	Proportions
$m_2^{\text{Initial}} > m_2^{\text{Final}}$	1,261,201	76.7%
$m_2^{\text{Initial}} < m_2^{\text{Final}}$	382,934	23.2%
$m_2^{\text{Initial}} = m_2^{\text{Final}}$	0	0.0%

Table 6.4: Comparison of m_2 of the initial failure state and the final state. m_2^{Initial}: m_2 of the initial state; m_2^{Final}: m_2 of the final state.

6.6 Additional Related Work

Additionally, our work has links to the following areas:

I. Edge Modification. Studies have shown that edge modification, such as adding, rewiring [110], or protecting some edges, can enhance network robustness. Our work theoretically connects edge removal with spectral moments.

II. Spectral Robustness. Wu and his colleagues propose natural connectivity, which can be regarded as the "average eigenvalue" of the adjacency matrix [125]. In our work, we look at the eigenvalue distribution of the random walk transition matrix via its spectral moments (equivalently, the spectral moments of the normalized Laplacian matrix).

6.7 Conclusion

We propose a spectral measure for network robustness: the second spectral moment m_2 of the random walk transition matrix. We study the m_2 as a spectral measure from (1) the assessment of the network robustness; (2) the design of networks with controlled robustness; (3) the behavioral study of a complex system through the observation of the evolution of its network robustness, under cascading failures. We theoretically and empirically demonstrate that m_2 can capture network robustness: *a graph with a smaller second spectral moment m_2 is more robust*. We show the relationship between m_2 and edge properties so that one can control the network robustness by manipulating its m_2 value.

For future work, we consider (1) studying the connection between higher order spectral moments and network robustness, e.g., the clustering coefficient is also a robustness measure as it captures the degree of transitivity of a network, and it is known to be lower bounded by the third spectral moment m_3 [15]; and (2) using the spectral moment predicting cascading failures.

Chapter 7

Summary and and Future Work

In this chapter, the main contributions of this dissertation are summarized, and the future research directions are also discussed here.

7.1 Summary

The dissertation proposes a compact, interpretable, visualizable, comparable and efficient representation of networks: *Network Shapes*, which represent any network with a 3D shape in the embedding space. We introduce two network shapes: *Kronecker hull* and *Spectral path*. We discuss how to build a network shape, how to interpret a network shape and the applications of network shapes.

In Chapter 2, we provide a general framework to build a network shape, which including three steps: (1) sample many subgraphs from the network; (2) map the network and its subgraphs to 3D vectors; and (3) fit a 3D shape to the set of 3D vectors. We introduce the first network shape: *Kronecker hull*, which utilizes random node sampling as the graph sampling method, Kronecker points as the embedding method, and convex hull as the shape fitting method. We show the interpretability of Kronecker points and Kronecker hulls in both theory and experiments. We study the characteristics of the Kronecker hulls of real-world networks. We demonstrate that Kronecker hulls can be used on applications such as network categorization and computing graph similarity.

In Chapter 3, we demonstrate that network shapes can be used to extend biometrics studies to network data, by solving two new problems in network studies: network identifi-

cation and network authentication. We propose two types of network identities, and we demonstrate their utility in solving both problems. The embedding-based identity is easy to construct, but the distribution-based identity utilizing network shapes performs better with simple methods. For network identification, we propose two techniques to predict the source of an anonymized graph, and for network authentication, we show that the supervised method yields a low equal error rate, and the Voronoi method enables controlling the false reject rate, while attaining a reasonable false accept rate across networks. We show that our graph-based methods can also be used for biometrics, authenticating users based on their touch data on devices.

In Chapter 4, we propose a spectral embedding space for networks, and we denote the 3D network embedding method, the *Spectral Point*, by using the truncated spectral moments of the network. Spectral points are interpretable as we prove that spectral moments are closely related to network structures such as edges, triads and squares, and the spectral moments are bounded by network properties such as the degree distribution and the global clustering coefficient. We derive the closed form of spectral points for various special graphs such as complete graphs, cycles, star graphs, complete bipartite graphs, and wheels. The experiments on real-world graphs show that their structure and properties identified in past literature are often captured by spectral points. Finally, we demonstrate that spectral moments can be used for network identification. The result shows that the spectral moments outperform the baselines and the truncated spectral moments do not lose much predictive power.

In Chapter 5, we introduce the second network shape: *Spectral Path*, which represent each network as a 3D spectral path in the embedding space, by connecting the expected spectral moments of the network and its subgraphs. We demonstrate the interpretability of spectral paths by investigating the *shapes* of spectral paths. We provide the theoretical relationship between the spectral moments of a network and those of its subgraphs. We show spectral paths can be used in network visualization, network identification and distinguishing

cospectral graphs.

In Chapter 6, we find that spectral moments, especially m_2, can capture the robustness of a network. More specifically, a graph with a smaller second spectral moment m_2 is more robust, for which We provid theoretical proofs and empirical results. We show that we can control the network robustness by manipulating its m_2 value, to design a network that is more robust under failures. Moreover, one can study how a complex networked system behaves under cascading failures by looking at how network robustness (m_2) evolves.

Next, we are going to look at the open questions about network shapes.

7.2 Future Research Directions

Here, we list some of the open questions from different perspectives.

7.2.1 Build a Network Shape

Sampling Subgraphs. In Kronecker hulls, we use random node sampling mainly because it is simple and fast. For spectral paths, we use random node sampling as the sampled subgraph can be viewed as a result of randomly removing nodes from a graph.

- If we use different sampling methods, how will the network shapes change? In Appendix A, we have a case study on other sampling methods such as *Random Edge Sampling* and *Random Walk Sampling*. A more systematic study may help us to have a better understanding.
- Can we learn the subgraphs that are the best representative for building or interpreting a network shape, towards specific applications or with extra information? For example, if a network is a node/edge attributed graph, one may consider sampling subgraphs with attribute information.

- Can we combine the graph sampling step with the network embedding step? For example, some network embedding methods utilize sampled subgraphs and substructures of a network.

Network Embedding. In Kronecker hulls, we are using Kronecker point as it captures the core-periphery structure of a network. In spectral paths, we use Spectral Points to capture networks structures/properties and the relationship between network and its subgraphs. The basic requirements of a proper embedding for a network shape includes visualizability, interpretability and carrying much structural information of the graph.

- Can we use other graph embeddings, and what are the critical properties of a proper embedding? Can we learn the embeddings based on different applications or objectives?

- Network shapes proposed in this thesis, mainly focus on the structural properties of a network and its subgraph. Can we add node/edge attribute information to the embeddings if attributed graphs are given?

Shape Fitting. In Kronecker hulls, we are using convex hull as the shape fitting as it is the most compact convex set covering all the points. In Spectral Paths, we utilize a path connecting the expected spectral points of subgraphs and that of the whole network.

- How about using other shapes (e.g., cubes, spheres) and what is their interpretability? We have a case study in Appendix A, and more systematic studies should be conducted. For spectral paths, we are using the expected spectral points of subgraphs with the same size. Can we include the variance/standard deviation of the spectral points?

- Network shapes do not reflect the density of the point distribution within the shape. Can we get the probability of a network having a subgraph on a certain point inside the shape? One idea is using density estimation to learn the function.

7.2.2 Applications

Graph Similarity. Can we connect the relationship between two network shapes (e.g., distance, overlap) with existing graph similarity measures such as graph edit distance or graph kernels?

Graph Classification. Graph classification is a problem with practical applications in many different domains, mainly to distinguish between graphs of different classes. Since network shapes capture the structural info of a network and its subgraphs, can we find proper features of network shapes, for graph classification use?

Network Models. For any network, we can compute its spectral moments. So, can we generate a random graph with given spectral moments? As spectral moments are closely related network structures and properties, a network model using spectral moments may provide synthetic graphs for benchmark.

Appendix A

Products of Network Shapes

A.1 Introduction

In this chapter, we designed various products for that enables researchers and practitioners to visualize their network data as customized 3D shapes. These products include a web platform WEBSHAPES. We provide a case study on real-world networks to explore the sensitivity of network shapes to different graph sampling, embedding, and fitting methods, and we show examples of understanding networks through their network shapes. Furthermore, we build a repository of precomputed network shapes for various networks.

A.2 WEBSHAPES

As we have mentioned, one can select various sampling methods (Step 1), embedding methods (Step 2), and fitting methods (Step 3) to build a network shape. Hence, we developed WEBSHAPES, a web platform for users to create customized network shapes for their need. Using WEBSHAPES, users can upload their network data or choose an existing dataset, select predefined methods and parameters (a sampling method, an embedding method, and a fitting method), and can visualize their network data as a 3D shape. Users can download the information of the network shape for further analysis. Figure A.1 is a screenshot of the platform. Next, we will briefly introduce the sampling methods, embedding methods, and fitting methods currently supported by WEBSHAPES.

(1) Three sampling methods:

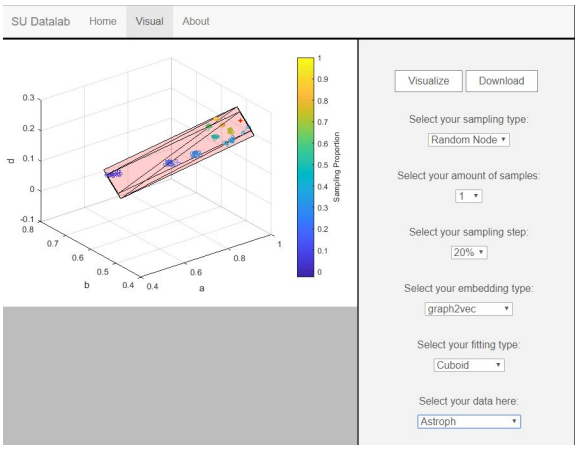


Figure A.1: WEBSHAPES Interface

- *Random Node Sampling*. Selects $p\%$ of nodes uniformly at random and outputs the subgraph induced by the selected nodes [37].

- *Random Edge Sampling*. Select $p\%$ of edges uniformly at random and outputs the subgraph induced by the selected edges [37].

- *Random Walk Sampling*. Starting from an initial random node performs a random walk to sample nodes. At each step, the random walk restarts from the initial node with probability (=0.15) [37]. The process continues until $p\%$ of nodes are sampled.

(2) Two embedding methods:

- *Kronecker Point* [12] is a 3D point (a, b, d) that embeds an undirected network, or any of its subgraphs, obtained using Stochastic Kronecker Graphs [38]. Values a, b and d are in range $[0, 1]$ and are closely related to the network structure. Generally, one can view a network as two groups of nodes and interpret a and d as the proportion of edges within each of the groups, and b as the fraction of edges between the two groups. Hence, we can split the whole embedding space into three regions based on a, b, d values: *Core-Periphery* ($a \geq b \geq d$), *Dual-Core* ($a \geq d \geq b$), and *Random* ($b \geq a \geq d$), and value a represents the *core strength* [12].

- *Graph2vec* views a graph as a document and the rooted subgraphs around each node as words. It extends document embedding neural networks to embed a graph as a vec-

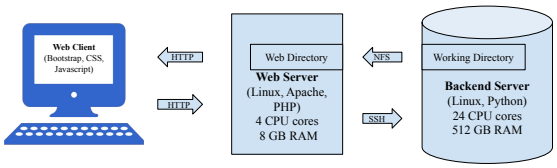


Figure A.2: Platform Architecture

tor [68]. For creating network shapes, we set the output dimension to 3.

(3) Three fitting methods:

- *Convex Hull* yields the smallest convex set that contains all 3D points given [39]. To save a convex hull, we save its boundary.
- *Cuboid* computes the minimal box (with right angles) around the 3D points. To save the cuboid, we save the 8 extreme points.
- *Sphere* uses the arithmetic mean of the points as the center and the longest distance from the points to the center as the radius to fit a sphere. To save it, we save the center and the radius.

After WEBSHAPES creates a network shape, users can download the figure and the files including boundary points (Convex Hull), extreme points (Cuboid) or the center and the radius (Sphere).

Architecture. Here, we briefly introduce the software architecture of the Web platform. The WEBSHAPES platform is composed of three components: the Web client (i.e., front-end), the Web server and the back-end server. The Web client is basically the webpages designed by the bootstrap framework using CSS and Javascript. Users can use the Web client to interact with the Web server through HTTP calls. The Web server uses the Linux stack, Apache Web server, and PHP. When the server receives a request to create a network shape, it will send a remote call to a strong back-end server. The back-end server creates a network shape and saves the figure file and related information of the shape in a directory, which is shared with the Web server using the Network File System (NFS). Figure A.2 illustrates the architecture and the servers configuration.

Table A.1: Shape Volumes for Sampling Methods ($\times 10^{-4}$)

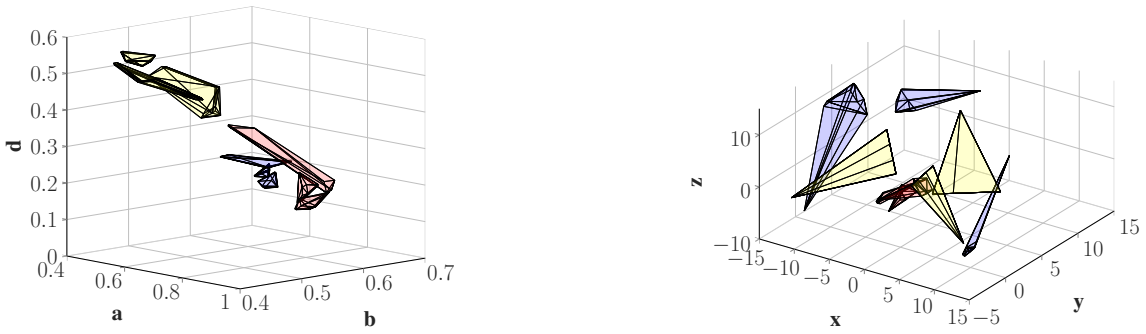
Type	Network	Random Node	Random Edge	Random Walk
Social Networks	Hyves	1.39	0.30	0.20
	MySpace	0.76	0.09	0.13
	YouTube	1.80	0.18	0.39
Collaboration Networks	Astro-Ph	1.55	4.22	0 (reduced to 2D)
	Cond-Mat	9.31	1.22	1.36
	Hep-Th	5.11	0.80	0.48
Road Networks	Road-CA	4.65	0.17	0.57
	Road-PA	3.65	3.83	0.26
	Road-tx	1.00	1.08	0.24

A.3 Case Study

Sampling methods. Table A.1 lists the volume of the network shapes created using different sampling methods. The result shows that shapes using Random Node sampling are in general large, while shapes using Random Walk sampling are generally small, and shapes using Random Edge sampling vary in volumes for different networks. As the volume captures the variance of the Kronecker points of the network and its subgraphs, it indicates that Random Node sampling generates samples with more variance, while Random Walk sampling generates more similar subgraphs. Figure A.3 provides the network shapes of Hyves and YouTube obtained using different sampling methods. It turns out that the three shapes intersect at the Kronecker point of the whole graph, and the Kronecker points of subgraphs are distributed in different directions. We can connect the observation to the sampling strategies: (1) The blue shapes have a much larger a value than others, as the Random Walk sampling (restarting with some probability from its initial node) prefers visiting the initial node and its near neighbors, i.e., 1-hop or 2-hop neighbors, which forms a dense core of the sample; (2) Though the yellow shapes and the red shapes are both in

(a) Hyves (b) YouTube

Figure A.3: Red: Random Node; Yellow: Random Edge; Blue: Random Walk



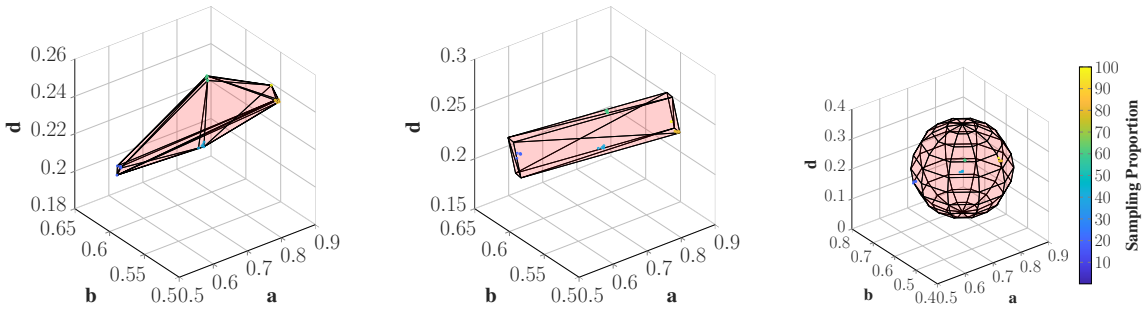
(a) Kronecker Point (b) graph2vec

Figure A.4: Red: Collaboration Networks; Yellow: Road Networks; Blue: Social Networks

the core-periphery region ($a \geq b \geq d$), the yellow shapes are above the red ones (having greater d value). This can be explained by the fact that Random Edge sampling has a slight bias towards high degree nodes and it generates denser samples than Random Node sampling does. Hence, for subgraphs sampled by Random Edge sampling, the group of periphery nodes have more internal connections.

Embedding methods. Figure A.4 provides the network shapes of the nine networks using Kronecker Point and *graph2vec* respectively. We have colored the shapes based on their network category. We observe (1) For Kronecker Point, as values a, b and d are all between 0 and 1, the whole embedding space is a $1 \times 1 \times 1$ cube; For *graph2vec*, the embedding values have no such bound; (2) When using Kronecker Point, the networks from the same category exhibit a clustering phenomenon. Moreover, social networks and collaboration networks are close while road networks are relatively far from others. This observation can be explained by previous results [12], i.e., most social networks and collaboration networks exhibit the core-periphery structure and these two categories have overlaps as they both involve human social behavior, but road networks often exhibit the dual-core structure. However, when using *graph2vec*, networks from different categories seem to form an onion-like structure: the collaboration networks are in the inside layer, social networks in the outside layer, and road networks are in the middle.

Fitting methods. Based on the definition of our three fitting methods, we know that for the



(a) Convex Hull (b) Cuboid (c) Sphere

Figure A.5: Comparison of Different Fitting Methods

Table A.2: Shape Volumes for Different Fitting Methods ($\times 10^{-4}$)

Type	Network	Convex Hull	Cuboid	Sphere
Social Networks	Hyves	0.30	1.15	673
	MySpace	0.09	0.30	189
	YouTube	0.18	0.61	344
Collaboration Networks	Astro-Ph	4.22	13.00	2398
	Cond-Mat	1.22	5.06	2246
	Hep-Th	0.80	2.18	1977
Road Networks	Road-CA	0.17	0.51	158
	Road-PA	3.83	13.00	3756
	Road-tx	1.08	3.59	2879

same set of points, $ConvexHull \subseteq Cuboid \subseteq Sphere$. Figure A.5 provides an example of the network shapes of Hyves. We list the volume of the shapes using different fitting methods in Table A.2. In general, we find that compared with other methods, convex hull is very compact, while sphere is very loose. Cuboid is in the middle, with 3 times volume of convex hull. As the process of building a network shape involves the graph sampling step, a good approximation of the embedding space of a network and its subgraphs should be both accurate and robust to sampling variance. Considering the trade-off, we believe convex hull and cuboid perform better in this case.

A.4 Network Shapes Repository

As computing network shapes can be computationally expensive and to enable researchers and practitioners to analyze those well-known public network data, we build a repository of precomputed network shapes for various networks. We collect 102 public networks

from 12 different categories. Furthermore, we generate various neural network architectures (graphs) for five different types of neural network families. The networks analyzed are mostly large, where our largest network have millions of nodes and edges. We build network shapes for each of these networks with different graph sampling, embedding, and shape fitting methods. Our repository that includes all the network shapes information is released on our website: (`http://b.link/shapesrepository`) and the code for generating network shapes is shared on Github. We provide a series of statistics of the network shapes and detailed analysis for understanding networks through their network shapes.

A.4.1 Large Network Data Collection

Our main sources for our large network data are four public repositories of networks: Network Repository [64], SNAP Datasets [43], Social Computing Data Repository at ASU [44], and AMiner [45]. In total, we collect 102 networks from 12 different categories. Following are the details of these network categories, with some examples of these types of networks. For a full list of networks and their statistics, please refer to the repository's website.

1. *Social Networks*: In social networks, nodes represent social network users and edges represent friendship or follower/followee relationships. We include thirteen social networks. Examples are MySpace [45] and YouTube [43].

2. *Biological Networks*: In our collection, biological networks mainly include protein-protein interaction networks. In particular, we include six biological networks. Examples are Bio-Grid-Yeast [64] and Bio-Human-Gene2 [64].

3. *Citation Networks*: Citation networks represent the references in papers (nodes in these networks). If a paper i cites paper j, the graph contains a directed edge from node i to j. In our study, as we consider undirected networks and citation networks often do

not have mutual connection (an edge from i to j and vice versa), we simply remove the direction of the edges. We include two citation networks, where an example is cit-DBLP [64].

4. *Collaboration Networks*: Collaboration networks represent scientific collaborations between scientists. In a collaboration network, an undirected edge between nodes i and j exists, if authors i and j have co-authored at least one paper. Our repository includes ten collaboration networks, e.g., Astro-Ph [43].

5. *Interaction Networks*: Interaction networks capture communications between individuals. Examples include an email network or a Wikipedia talk pages. For a Wikipedia talk page network, an undirected edge indicates that user A and user B made edits on each others' talk page. We include ten interaction networks, e.g., ia-email-EU [64].

6. *Road Networks*: In road networks, nodes are intersections/endpoints and undirected edges are the roads connecting these intersections/road endpoints. In our repository, we analyze five road networks, where an example is Road-CA [43].

7. *Web Networks*: In web networks, nodes represent web pages and directed edges represent hyperlinks between them. In our study, we simply remove the direction of the edges. We include four web networks, e.g., web-Google [43].

8. *Recommendation Networks*: Recommendation networks are often bipartite networks, representing users rating items with the rating scores being the edge weights. In our study, we consider unweighted networks, so we remove the weight information. As long as a user has rated an item, there is an edge between the user and the item. We include two of recommendation networks, where an example is the Rec-Github [64] network.

9. *Economic Networks*: An economic network represents a set of individuals or groups interacting (e.g. trading) for profit. We include only one economic networks: Econ-

Table A.3: Statistics for Neural Network Datasets

Neural Network Type	# Layers	Nodes	Edges	Average Degree	Density
Recurrent Neural Network	4	2,000	750,000	750.0	0.375
Deep Convolutional Network	8	1,659	166,813	201.1	0.121
Deep Belief Network	7	2,750	750,000	545.5	0.198
Autoencoder	3	2,000	640,000	640.0	0.320
Deep Fully-connected Network	7	2,365	984,578	832.6	0.352

Poli-Large [64].

10. *Heterogeneous Networks*: Heterogeneous have different types of nodes, for example users and groups, and edges often denote membership relations (a user is a member of a group). We analyze eight heterogenous networks, e.g., actor [64]

11. *Labeled Networks*: Labeled networks have labels on their nodes or edges. We include 36 labeled networks. Example: AIDS [64].

12. *Miscellaneous Networks*: These networks cannot be classified under the above-mentioned categories. We include five such networks, where an example is 3Dspectralwave [64]

As these network datasets are collected from different platforms, they follow different formats; hence, we transform each of the them to a unified tab-delimited format, which can be downloaded from the repository's website.

In addition to the above-mentioned networks, recent studies have shown the success of deep neural networks in many applications [126]. With many recently proposed deep neural network architectures [127, 128], there is a strong need to study these architectures in-depth. A neural network architecture can be represented as a graph; hence, we generate five different neural network architectures for the repository: Recurrent Neural Networks (RNN) [129], Deep Convolutional Networks [130], Deep Belief Networks [131], Autoencoders [132], and Deep Fully-connected Networks [130]. Table A.3 provides the details for the generated networks.

A.5 Conclusion

In this chapter, we first present WEBSHAPES, a web platform implementing the *network shape* framework which allows users to visualize their network data as 3D shapes with different methods. We have a case study on networks from different categories. We demonstrate that the properties of different graph sampling methods can be connected to the network shape; with different embedding methods networks have individual network shapes; convex hull and cuboid provide a better approximation of the embedding space of a network from the view of the accuracy and robustness trade-off. As the network shape framework is flexible, we can integrate more new graph sampling methods, embedding methods and fitting methods in WEBSHAPES in the future.

Moreover, we provide the first network data repository with network embeddings and 3D network representations, NETWORK SHAPES REPOSITORY, to the best of our knowledge. For visualizing and downloading network shapes and network embeddings, we build a website and we share the code for building network shapes to public. To build the network shapes repository, we collect 102 public networks from 12 different categories. We have transformed the network data into a consistent unified format and all the network data can be downloaded from the repository. We generate graphs representing five different neural network families as public network data for research.

Bibliography

[1] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017, pp. 1025–1035.

[2] L. Backstrom and J. Leskovec, "Supervised random walks: predicting and recommending links in social networks," in *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 2011, pp. 635–644.

[3] S. Jin and R. Zafarani, "Emotions in social networks: Distributions, patterns, and models," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 1907–1916.

[4] M. Newman, *Networks*. Oxford university press, 2018.

[5] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD conference*. ACM, 2016, pp. 855–864.

[6] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.

[7] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 1067–1077.

[8] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *arXiv preprint arXiv:1709.05584*, 2017.

[9] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecu-

lar fingerprints," in *Advances in neural information processing systems*, 2015, pp. 2224–2232.

[10] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems*, 2016, pp. 3844–3852.

[11] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[12] S. Jin and R. Zafarani, "Representing networks with 3d shapes," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 177–186.

[13] S. Jin, V. V. Phoha, and R. Zafarani, "Network identification and authentication," in *2019 IEEE International Conference on Data Mining (ICDM). IEEE*, 2019.

[14] S. Jin, R. Wituszynski, M. Caiello-Gingold, and R. Zafarani, "Webshapes: Network visualization with 3d shapes," in *Proceedings of the 13th International Conference on Web Search and Data Mining*, 2020, pp. 837–840.

[15] S. Jin and R. Zafarani, "The spectral zoo of networks: Embedding and visualizing networks with spectral moments," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1426–1434.

[16] R. Abdolazimi, S. Jin, and R. Zafarani, "Noise-enhanced community detection," in *Proceedings of the 31st ACM Conference on Hypertext and Social Media*, 2020, pp. 271–280.

[17] S. Jin, H. Tian, J. Li, and R. Zafarani, "A spectral representation of networks: The path of subgraphs," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 698–708.

[18] S. Jin, R. Ma, J. Li, S. Eftekharnejad, and R. Zafarani, "A spectral measure for network robustness: Assessment, design, and evolution," in *202 IEEE International Conference on Knowledge Graphs (ICKG). IEEE*, 2022.

[19] S. Jin, V. V. Phoha, and R. Zafarani, "Graph-based identification and authentication: A stochastic kronecker approach," *IEEE Transactions on Knowledge & Data Engineering*, vol. 34, no. 07, pp. 3282–3294, 2022.

[20] R. Tommasini, S. Basu Roy, X. Wang, H. Wang, H. Ji, J. Han, P. Nakov, G. Da San Martino, F. Alam, M. Schedl *et al.*, "Accepted tutorials at the web conference 2022," in *Companion Proceedings of the Web Conference 2022*, 2022, pp. 391–399.

[21] S. Jin and R. Zafarani, "Sentiment prediction in social networks," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2018, pp. 1340–1347.

[22] X. Zhou, S. Jin, and R. Zafarani, "Sentiment paradoxes in social networks: Why your friends are more positive than you?" in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 14, 2020, pp. 798–807.

[23] R. Ma, S. Jin, S. Eftekharnejad, R. Zafarani, and W. P. J. Philippe, "A probabilistic cascading failure model for dynamic operating conditions," *IEEE Access*, vol. 8, pp. 61 741–61 753, 2020.

[24] J. Li, T. Zhang, H. Tian, S. Jin, M. Fardad, and R. Zafarani, "Sgcn: A graph sparsifier based on graph convolutional networks," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2020, pp. 275–287.

[25] J. Li, T. Zhang, S. Jin, M. Fardad, and R. Zafarani, "Adversparse: An adversarial attack framework for deep spatial-temporal graph neural networks," in *ICASSP*

2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2022, pp. 5857–5861.

[26] J. Li, T. Zhang, H. Tian, S. Jin, M. Fardad, and R. Zafarani, "Graph sparsification with graph convolutional networks," *International Journal of Data Science and Analytics*, vol. 13, no. 1, pp. 33–46, 2022.

[27] Z. Zhang, M. Wang, Y. Xiang, Y. Huang, and A. Nehorai, "Retgk: Graph kernels based on return probabilities of random walks," *arXiv preprint arXiv:1809.02670*, 2018.

[28] K. Dong, A. R. Benson, and D. Bindel, "Network density of states," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1152–1161.

[29] L. Huang, A. J. Graven, and D. Bindel, "Density of states graph kernels," in *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*. SIAM, 2021, pp. 289–297.

[30] A. Tsitsulin, D. Mottin, P. Karras, A. Bronstein, and E. Müller, "Netlsd: hearing the shape of a graph," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2347–2356.

[31] Y. Liu, T. Safavi, A. Dighe, and D. Koutra, "Graph summarization methods and applications: A survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, pp. 1–34, 2018.

[32] D. Koutra, U. Kang, J. Vreeken, and C. Faloutsos, "Vog: Summarizing and understanding large graphs," in *Proceedings of the 2014 SIAM international conference on data mining*. SIAM, 2014, pp. 91–99.

[33] Y. Koren, "On spectral graph drawing," in *International Computing and Combinatorics Conference*. Springer, 2003, pp. 496–508.

[34] D. Spielman, "Spectral graph theory," *Combinatorial scientific computing*, vol. 18, 2012.

[35] B. Bollobás, "Almost every graph has reconstruction number three," *Journal of Graph Theory*, vol. 14, no. 1, pp. 1–4, 1990.

[36] T. Von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. J. van Wijk, J.-D. Fekete, and D. W. Fellner, "Visual analysis of large graphs: state-of-the-art and future research challenges," in *Computer graphics forum*, vol. 30, no. 6. Wiley Online Library, 2011, pp. 1719–1749.

[37] J. Leskovec and C. Faloutsos, "Sampling from large graphs," in *Proceedings of the SIGKDD*. ACM, 2006, pp. 631–636.

[38] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, "Kronecker graphs: An approach to modeling networks," *JMLR*, vol. 11, no. Feb, pp. 985–1042, 2010.

[39] M. De Berg, M. Van Kreveld, M. Overmars, and O. C. Schwarzkopf, "Computational geometry," in *Computational geometry*. Springer, 2000, pp. 1–17.

[40] F. P. Preparata and S. J. Hong, "Convex hulls of finite sets of points in two and three dimensions," *Communications of the ACM*, vol. 20, no. 2, pp. 87–93, 1977.

[41] J. Leskovec and C. Faloutsos, "Scalable modeling of real graphs using kronecker multiplication," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 497–504.

[42] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, pp. 469–483, 1996.

[43] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.

[44] R. Zafarani and H. Liu, "Social computing data repository at ASU," 2009. [Online]. Available: http://socialcomputing.asu.edu

[45] Y. Zhang, J. Tang, Z. Yang, J. Pei, and P. S. Yu, "Cosnet: Connecting heterogeneous social networks with local and global consistency," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1485–1494.

[46] M. E. Newman, "The structure and function of complex networks," *SIAM review*, vol. 45, no. 2, pp. 167–256, 2003.

[47] S. P. Borgatti and M. G. Everett, "Models of core/periphery structures," *Soc. networks*, vol. 21, no. 4, pp. 375–395, 2000.

[48] P. Erdös and A. Rényi, "On random graphs, i," *Publicationes Mathematicae (Debrecen)*, vol. 6, pp. 290–297, 1959.

[49] F. Luo, B. Li, X.-F. Wan, and R. H. Scheuermann, "Core and periphery structures in protein interaction networks," in *Bmc Bioinformatics*, vol. 10, no. 4. BioMed Central, 2009, p. S8.

[50] D. S. Bassett, N. F. Wymbs, M. P. Rombach, M. A. Porter, P. J. Mucha, and S. T. Grafton, "Task-based core-periphery organization of human brain dynamics," *PLoS computational biology*, vol. 9, no. 9, p. e1003171, 2013.

[51] M. Bastian, S. Heymann, M. Jacomy *et al.*, "Gephi: an open source software for exploring and manipulating networks." *Icwsm*, vol. 8, pp. 361–362, 2009.

[52] J. Zhang, M. S. Ackerman, and L. Adamic, "Expertise networks in online communities: Structure and algorithms," in *Proceedings of the 16th International*

Conference on World Wide Web, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 221–230. [Online]. Available: http://doi.acm.org/10.1145/1242572.1242603

[53] T. Feder and R. Motwani, "Clique partitions, graph compression and speeding-up algorithms," in *Proceedings of the twenty-third annual ACM symposium on Theory of computing*. ACM, 1991, pp. 123–133.

[54] A. Apostolico and G. Drovandi, "Graph compression by bfs," *Algorithms*, vol. 2, no. 3, pp. 1031–1044, 2009.

[55] C. A. Packer and L. B. Holder, "Graphzip: Dictionary-based compression for mining graph streams," *arXiv preprint arXiv:1703.08614*, 2017.

[56] B. Zhou, J. Pei, and W. Luk, "A brief survey on anonymization techniques for privacy preserving publishing of social network data," *ACM Sigkdd Explorations Newsletter*, vol. 10, no. 2, pp. 12–22, 2008.

[57] L. Backstrom, C. Dwork, and J. Kleinberg, "Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 181–190.

[58] S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of STOC*. ACM, 1971, pp. 151–158.

[59] A. K. Jain, L. Hong, S. Pankanti, and R. Bolle, "An identity-authentication system using fingerprints," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1365–1388, 1997.

[60] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of fingerprint recognition*. Springer Science & Business Media, 2009.

[61] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt, "Graph kernels," *Journal of Machine Learning Research*, vol. 11, no. Apr, pp. 1201–1242, 2010.

[62] T. Gärtner, P. Flach, and S. Wrobel, "On graph kernels: Hardness results and efficient alternatives," in *Learning theory and kernel machines*. Springer, 2003, pp. 129–143.

[63] D. F. Gleich and A. B. Owen, "Moment-based estimation of stochastic kronecker graph parameters," *Internet Mathematics*, vol. 8, no. 3, pp. 232–256, 2012.

[64] R. Rossi and N. Ahmed, "The network data repository with interactive graph analytics and visualization." in *AAAI*, vol. 15, 2015, pp. 4292–4293.

[65] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial tessellations: concepts and applications of Voronoi diagrams*. John Wiley & Sons, 2009, vol. 501.

[66] D. Koutra, A. Parikh, A. Ramdas, and J. Xiang, "Algorithms for graph similarity and subgraph matching," in *Proc. Ecol. Inference Conf*, 2011.

[67] D. Cohen-Steiner, W. Kong, C. Sohler, and G. Valiant, "Approximating the spectrum of a graph," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1263–1271.

[68] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," *arXiv preprint arXiv:1707.05005*, 2017.

[69] D. P. Huttenlocher, W. J. Rucklidge, and G. A. Klanderman, "Comparing images using the hausdorff distance under translation," in *Proceedings of CVPR*. IEEE, 1992, pp. 654–656.

[70] A. Serwadda, V. V. Phoha, and Z. Wang, "Which verifiers work?: A benchmark evaluation of touch-based authentication algorithms," in *2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*. IEEE, 2013, pp. 1–8.

[71] J. R. Ullmann, "Bit-vector algorithms for binary constraint satisfaction and subgraph isomorphism," *Journal of Experimental Algorithmics (JEA)*, vol. 15, pp. 1–6, 2010.

[72] A. K. Jain and S. Z. Li, *Handbook of face recognition*. Springer, 2011.

[73] W. W. Boles and B. Boashash, "A human identification technique using images of the iris and wavelet transform," *IEEE transactions on signal processing*, vol. 46, no. 4, pp. 1185–1188, 1998.

[74] J. Mantyjarvi, M. Lindholm, E. Vildjiounaite, S.-M. Makela, and H. Ailisto, "Identifying users of portable devices from gait pattern with accelerometers," in *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, vol. 2. IEEE, 2005, pp. ii–973.

[75] G. Li, M. Semerci, B. Yener, and M. J. Zaki, "Effective graph classification based on topological and label attributes," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 5, no. 4, pp. 265–283, 2012.

[76] L. Ralaivola, S. J. Swamidass, H. Saigo, and P. Baldi, "Graph kernels for chemical informatics," *Neural networks*, vol. 18, no. 8, pp. 1093–1110, 2005.

[77] J. P. Canning, E. E. Ingram, S. Nowak-Wolff, A. M. Ortiz, N. K. Ahmed, R. A. Rossi, K. R. Schmitt, and S. Soundarajan, "Predicting graph categories from structural properties," *arXiv preprint arXiv:1805.02682*, 2018.

[78] F. R. Chung and F. C. Graham, *Spectral graph theory*. American Mathematical Soc., 1997, no. 92.

[79] A. J. Hoffman, "On eigenvalues and colorings of graphs," in *Selected Papers Of Alan J Hoffman: With Commentary*. World Scientific, 2003, pp. 407–419.

[80] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in neural information processing systems*, 2002, pp. 849–856.

[81] A. D. Broido and A. Clauset, "Scale-free networks are rare," *Nature communications*, vol. 10, no. 1, pp. 1–10, 2019.

[82] A.-L. Barabási and E. Bonabeau, "Scale-free networks," *Scientific american*, vol. 288, no. 5, pp. 60–69, 2003.

[83] E. Estrada, "Characterization of the folding degree of proteins," *Bioinformatics*, vol. 18, no. 5, pp. 697–704, 2002.

[84] X. Chen and J. Qian, "Bounds on the number of closed walks in a graph and its applications," *Journal of Inequalities and Applications*, vol. 2014, no. 1, pp. 1–9, 2014.

[85] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," *ACM SIGCOMM computer communication review*, vol. 29, no. 4, pp. 251–262, 1999.

[86] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman, "Search in power-law networks," *Physical review E*, vol. 64, no. 4, p. 046135, 2001.

[87] X. Zhang, T. Martin, and M. E. Newman, "Identification of core-periphery structure in networks," *Physical Review E*, vol. 91, no. 3, p. 032803, 2015.

[88] S. Milgram, "The small world problem," *Psychology today*, vol. 2, no. 1, pp. 60–67, 1967.

[89] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *nature*, vol. 393, no. 6684, p. 440, 1998.

[90] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow, "The anatomy of the facebook social graph," *arXiv preprint arXiv:1111.4503*, 2011.

[91] S.-H. Yook, Z. N. Oltvai, and A.-L. Barabási, "Functional and topological characterization of protein interaction networks," *Proteomics*, vol. 4, no. 4, pp. 928–942, 2004.

[92] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabási, "The large-scale organization of metabolic networks," *Nature*, vol. 407, no. 6804, pp. 651–654, 2000.

[93] K. Tian, W. Kong, and G. Valiant, "Learning populations of parameters," in *Advances in neural information processing systems*, 2017, pp. 5778–5787.

[94] L. N. Trefethen, *Approximation theory and approximation practice*. Siam, 2013, vol. 128.

[95] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: simple building blocks of complex networks," *Science*, vol. 298, no. 5594, pp. 824–827, 2002.

[96] V. M. Preciado and A. Jadbabaie, "Moment-based spectral analysis of large-scale networks using local structural information," *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, pp. 373–382, 2012.

[97] V. M. Preciado, A. Jadbabaie, and G. C. Verghese, "Structural analysis of laplacian spectral properties of large-scale networks," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2338–2343, 2013.

[98] S. Verma and Z.-L. Zhang, "Hunt for the unique, stable, sparse and fast feature learning on graphs," in *Advances in Neural Information Processing Systems*, 2017, pp. 88–98.

[99] I. Pinelis, "Relations between the first four moments," *arXiv preprint arXiv:1111.6220*, 2011.

[100] A. J. Schwenk, "Almost all trees are cospectral," *New directions in the theory of graphs*, pp. 275–307, 1973.

[101] R. C. Wilson and P. Zhu, "A study of graph spectra for comparing graphs and trees," *Pattern Recognition*, vol. 41, no. 9, pp. 2833–2841, 2008.

[102] P. J. Kelly *et al.*, "A congruence theorem for trees." *Pacific Journal of Mathematics*, vol. 7, no. 1, pp. 961–968, 1957.

[103] S. M. Ulam, *A collection of mathematical problems*. Interscience Publishers, 1960, vol. 8.

[104] P. V. O'Neil, "Ulam's conjecture and graph reconstructions," *The American Mathematical Monthly*, vol. 77, no. 1, pp. 35–43, 1970.

[105] S. Butler and J. Grout, "A construction of cospectral graphs for the normalized laplacian," *arXiv preprint arXiv:1008.3646*, 2010.

[106] J. R. Magnus and H. Neudecker, *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons, 2019.

[107] S. K. Butler, "Eigenvalues and structures of graphs," Ph.D. dissertation, UC San Diego, 2008.

[108] W. Ellens and R. E. Kooij, "Graph measures and network robustness," *arXiv:1311.5064*, 2013.

[109] S. Freitas, D. Yang, S. Kumar, H. Tong, and D. H. Chau, "Graph vulnerability and robustness: A survey," *arXiv preprint arXiv:2105.00419*, 2021.

[110] A. Beygelzimer, G. Grinstein, R. Linsker, and I. Rish, "Improving network robustness by edge modification," *Physica A*, vol. 357, no. 3-4, pp. 593–612, 2005.

[111] F. Chung and L. Lu, "Connected components in random graphs with given expected degree sequences," *Annals of combinatorics*, vol. 6, no. 2, pp. 125–145, 2002.

[112] F. Chung, L. Lu, and V. Vu, "The spectra of random graphs with given expected degrees," *Internet Mathematics*, vol. 1, no. 3, pp. 257–275, 2004.

[113] F. Chung and L. Lu, "The average distances in random graphs with given expected degrees," *PNAS*, vol. 99, no. 25, pp. 15 879–15 882, 2002.

[114] H. Tong, B. A. Prakash, C. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, and D. H. Chau, "On the vulnerability of large graphs," in *ICDM*. IEEE, 2010, pp. 1091–1096.

[115] P. Van Mieghem, D. Stevanović, F. Kuipers, C. Li, R. Van De Bovenkamp, D. Liu, and H. Wang, "Decreasing the spectral radius of a graph by link removals," *Physical Rev. E*, vol. 84, no. 1, p. 016101, 2011.

[116] F. Chung, L. Lu, and V. Vu, "Eigenvalues of random power law graphs," *Annals of Combinatorics*, vol. 7, no. 1, pp. 21–33, 2003.

[117] P. Erdos, A. Rényi *et al.*, "On the evolution of random graphs," *Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960.

[118] S. Janson, D. E. Knuth, T. Łuczak, and B. Pittel, "The birth of the giant component," *Random Structures & Algorithms*, vol. 4, no. 3, pp. 233–358, 1993.

[119] M. Molloy and B. Reed, "A critical point for random graphs with a given degree sequence," *Random structures & algorithms*, vol. 6, no. 2-3, pp. 161–180, 1995.

[120] ——, "The size of the giant component of a random graph with a given degree sequence," *Combinatorics probability and computing*, vol. 7, no. 3, pp. 295–305, 1998.

[121] B. Bollobás, "The evolution of random graphs," *Trans. of the AMS*, vol. 286, no. 1, pp. 257–274, 1984.

[122] R. R. Williams, "Faster all-pairs shortest paths via circuit complexity," *SIAM Journal on Computing*, vol. 47, no. 5, pp. 1965–1985, 2018.

[123] A.-L. Barabási, "Network science," *Philosophical Trans. of the Royal Soc. A*, vol. 371, no. 1987, p. 20120375, 2013.

[124] R. Christie and I. Dabbagchi, "30 bus power flow test case," 1993.

[125] J. Wu, M. Barahona, Y.-J. Tan, and H.-Z. Deng, "Spectral measure of structural robustness in complex networks," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 41, no. 6, pp. 1244–1252, 2011.

[126] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[127] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[128] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[129] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[130] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[131] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[132] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length and helmholtz free energy," in *Advances in neural information processing systems*, 1994, pp. 3–10.

Vita

Name of Author: Shengmin Jin

Place of Birth: Shanghai, China

Major: Computer/Information Science and Engineering

Education

- Ph.D., Syracuse University, Syracuse, USA, 2022
- M.S., Syracuse University, Syracuse, USA, 2016
- B.S., Fudan University, Shanghai, China, 2010

Professional Experience

- Research Assistant, Syracuse University, Syracuse, USA, August 2016 - December 2022
- Applied Scientist Intern, Amazon.com, Seattle, USA, May 2020 - August 2020
- Research Intern, University of Rochester, Rochester, USA, May 2015 - August 2015
- Infrastructure Specialist, IBM, Shanghai, China, March 2013 - August 2014
- Software Engineer, eBay, Shanghai, China, April 2010 - March 2013

Teaching Experience (Assistant)

- Objective Oriented Design, September 2022 - December 2022

- Introduction to Analysis of Algorithms, September 2020 - December 2020
- Social Media Mining, January 2020 - May 2020
- Introduction to Analysis of Algorithms, September 2019 - December 2019
- Structured Programming and Formal Method, January 2016 - May 2016
- Mathematical Basis for Computing Science, September 2016 - December 2016

Awards and Honors

- KDD Student Travel Grant, 2022
- KDD Student Registration Award, 2020
- Syracuse University ECS Research Day Best Poster, 2020
- IEEE ICDM Student Travel Award, 2019
- Syracuse University Travel Grant, 2019
- IEEE ICDM Student Travel Award, 2018
- Syracuse University Travel Grant, 2018
- CIKM Student Travel Grant, 2017
- Syracuse University Travel Grant, 2017
- Outstanding Graduate Student in Computer Science, 2016 (awarded only to one student in the college of engineering)