Investigating the relationship between Eigenvalues and Graph Properties

Linear Algebra Project

Team – 85

Shaunak Biswas (2022111024)

Nitin Avuthu (2022115002)

Gargi Shroff (2022114009)

1. Introduction

The profound relationship between eigenvalues and graph properties provides us with an insight into the intricate nature of graphs. Eigenvalues, arising from the adjacency matrix and Laplacian matrix of a graph, serve as powerful tools for understanding graph characteristics.

The spectrum of eigenvalues offers a glimpse into the connectivity of a graph, revealing its underlying structural framework. By examining the number of zero eigenvalues in the Laplacian matrix, one can discern the presence and count of connected components within the graph.

Furthermore, the distribution and magnitude of eigenvalues furnish valuable clues about the existence of clusters and substructures of graphs. Beyond their descriptive power, eigenvalues have practical implications, aiding in graph partitioning, community detection, and spectral graph theory.

The analysis of eigenvalues across graphs with diverse attributes allows us to uncover the intricate interplay between eigenvalues and graph properties. Through this exploration, we gain a deeper understanding of the fundamental disciplines of graph theory and linear algebra.

1.1 Graph Theory and associated definitions:

The branch of Mathematics aimed at studying the properties and relationships of graphs, Graph Theory aims to model and analyse real-world systems such as social or biological networks as well as transportation networks.

For this project, we will define a graph as a set of vertices V connected by a set of edges E.

$$G = (V, E)$$

where $(u, v) \in E$ if $\exists$ an edge between u and v where $u, v \in V$

1.1.1 Directed Graph

In a directed graph, edges are given a direction that denotes a one-way connection between vertices. A directed graph has two distinct degree measures for each vertex: indegree and outdegree. The number of edges arriving into a vertex is known as indegree, and the number of edges leaving a vertex is known as outdegree.

1.1.2 Undirected Graph

Undirected graphs have edges without a have a direction. The edges indicate a two-way relationship, in that each edge can be traversed in both directions. The degree of a vertex in an undirected graph is given by the number of edges associated with that vertex. Every undirected graph can be thus imagined as a directed graph with each edge being represented as two edges, one from u to v and on from v to u.

1.2 Eigenvalue and Eigenvector

For a matrix, A, a vector which on being multiplied by A gives a scalar multiple of itself is called an eigenvector x of A. The scalar multiple, denoted by λ is known as the corresponding eigenvalue.

$$Ax = \lambda x$$

1.3 Representation of Graphs using Linear Algebra

1.3.1 The Adjacency Matrix

One common method to represent graphs other than the more commonly used Adjacency List is the Adjacency matrix, typically used for dense graphs. It gives us the ability to perform matrix operations and analyse graph properties using linear algebra techniques.

For a graph G, the adjacency matrix A = A(G) {is denoted by an N x N square matrix such that each entry A_{ij} is equal to the number of edges

connecting vertices V_i to V_j for non-diagonal elements. For diagonal elements in an undirected graph, A_{ii} denotes twice the number of self-loops on vertex V_i while denoting the number of self-loops on vertex V_i in directed graphs.

For an undirected graph,

- A is symmetric $\because$ if $\exists$ an edge $(u, v) \in E$, $\exists$ a vertex from v to u.
- The eigenvalues of A are real, following from the previous conclusion.

For simple graphs,
- $(u, u) \notin E \; \forall \; u \in V$, as self-loops are absent.

- $A_{ij} = \begin{cases} 1, & (I, j) \in E \\ 0, & \text{otherwise} \end{cases}$

1.3.2 The Laplacian Matrix

Also known as the combinatorial Laplacian of G, $L = L(G)$ is given by

$$L(G) = D(G) - A(G)$$

where $D = D(G)$ represents the diagonal matrix containing the degrees of the vertices of G.

For a simple graph, the Laplacian Matrix can be represented as follows:

$$L_{ij} = \begin{cases} \deg(I), & i = j \\ -A_{ij}, & i \neq j \end{cases}$$

Note
- Unless otherwise specified, the eigenvalues of a graph are used to refer to the eigenvalues of its adjacency matrix.
- The eigenvalues of a graph G are represented as λ_i, $i = 1, 2, \ldots, n$, where,

$$\lambda_1 \geq \lambda_2 \geq \cdot\cdot\cdot \geq \lambda_k \geq \lambda_{k+1} \geq \cdot\cdot\cdot \geq \lambda_n$$

- The eigenvalues of the Laplacian Matrix of the graph are represented as μ_i, i = 1, 2, …, n, where,

$$\mu_1 \leq \mu_2 \leq \cdot\cdot\cdot \leq \mu_k \leq \mu_{k+1} \leq \cdot\cdot\cdot \leq \mu_n$$

- Unless otherwise specified, we consider simple graphs throughout the discourse.

2. State of the Art

2.1 Graph Fourier Transform Centrality for Taipei Metro System
Chien-Cheng, Tsen Su-Ling Lee, IS3C 2020

This paper aims to understand the centrality of stations in the Taipei Metro System, represented as a graph representation of the subway network of Taipei. Various centralities have been employed to understand the influence of nodes in the network.


The route map of Taipei metro system. There are 120 stations in this system.

The GFT is an extension of the classical Fourier Transform to graph-structured data. It allows for analysing signals on graphs in the frequency domain. In this case, the graph signal represents the importance of each station.

The Graph Laplacian has been computed to quantitively understand the connectivity of the graph in order to perform the GFT. The graph

Laplacian is decomposed into eigenvalues and eigenvectors, which form the basis for the GFT.

A centrality measure called Graph Fourier Transform Centrality, derived from the GFT is introduced which quantifies the centrality of each station based on its frequency components in the GFT domain.
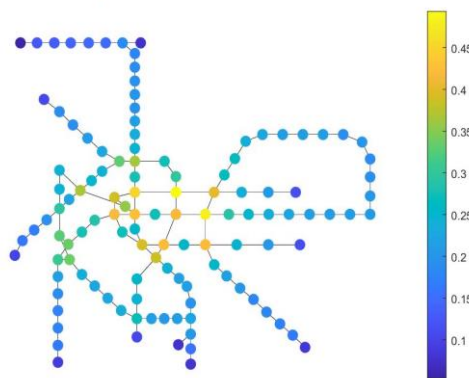
The colour representation of the graph Fourier Transform centrality (GFTC) measure of Taipei metro system.

Higher GFTC values indicate stations with higher centrality or influence in the metro system. The central stations are identified through this serving as an indicator of the major transfer hubs or the stations which play critical roles in connecting different regions of the network.

Comparing to the results from the degree centrality and the betweenness centrality, the paper validates the use for GFTC to analyse the centrality of the nodes in the Taipei Metro Station.
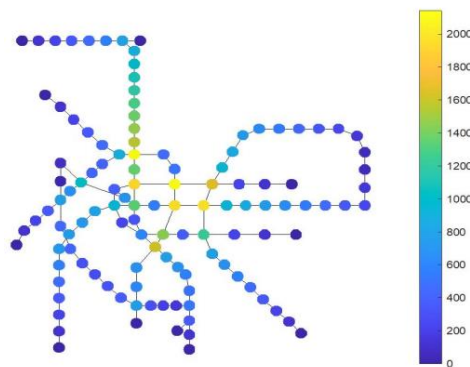
The colour representation of the betweenness centrality measure of Taipei metro system

This paper employs linear algebra techniques, specifically eigenvalues and eigenvectors of the graph Laplacian, to perform the Graph Fourier Transform and assess the centrality of stations.

Building up on this paper, one could identify communities or clusters of stations within the metro system by investigating how the GFTC values can assist in community detection algorithms, providing insights into groups of stations that exhibit similar centrality patterns or serve specific functions within the network.

2.2 Resilience Notions for Scale-free Networks

The research paper titled "Resilience Notions for Scale-free Networks" by Erkel and Matta focuses on the resilience analysis of scale-free networks. It explores various resilience notions and metrics to understand and quantify the robustness of scale-free networks against failures and attacks. It provides valuable insights for designing more robust and resilient networks in real-world applications.

2.3 Algebraic Connectivity of Graphs

 "Algebraic Connectivity of Graphs" by F. R. K. Chung: This influential paper investigates the concept of algebraic connectivity, defined as the second smallest eigenvalue of the Laplacian matrix. The paper provides insights into the importance of the eigenvalues in characterizing graph structures.

3. Connectivity

Connectivity refers to the degree to which nodes or vertices in a graph are interconnected or can be reached from one another through paths or edges. A graph is said to be connected if there exists a path between any two nodes in the graph.
The concept of connectivity captures the notion of how easily information or influence can flow through the network. It is a fundamental property of graphs that plays a crucial role in understanding various graph properties, such as network robustness, communication efficiency, and the ability to transmit signals or resources between nodes.

We aim to analyse the connectivity of graphs using its adjacency and eigenvalues.

3.1 Bipartite Graphs

For a graph G= (V, E), if the set of its vertices V can be partitioned into to disjoint sets V1 and V2 such that every edge in E connects a vertex from V1 to V2 is called a bipartite graph. This implies that no two vertices in one set are connected.

Statement: G is bipartite if and only if all eigenvalues of graph are symmetric about zero, i.e., for all eigenvalues λ, - λ is also an eigenvalue of the same graph.

Proof:

$\Leftarrow$

For a bipartite graph G, its vertices can be rearranged such that

$$A(G) = \begin{bmatrix} 0 & B \\ B^T & 0 \end{bmatrix}$$

Let v = $\begin{bmatrix} x \\ y \end{bmatrix}$ be an eigenvector of A with eigenvalue λ. Then we have,

$$A(G) \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \lambda \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\Rightarrow \quad \begin{bmatrix} B\,y \\ B^T x \end{bmatrix} = \begin{bmatrix} \lambda\,x \\ \lambda\,y \end{bmatrix}$$

Thus, we have $B\,y = \lambda\,x$ and $B^T x = \lambda\,y$. From this we have,

$$\begin{bmatrix} 0 & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ -y \end{bmatrix} = \begin{bmatrix} -B\,y \\ B^T x \end{bmatrix} = \begin{bmatrix} -\lambda\,x \\ \lambda\,y \end{bmatrix} = -\lambda \begin{bmatrix} x \\ -y \end{bmatrix}$$

From the above equation, we can conclude that $-\lambda$ is an eigenvalue for the same matrix A(G) $\begin{bmatrix} \lambda\,x \\ \lambda\,y \end{bmatrix}$ corresponding to the eigenvector $\begin{bmatrix} x \\ -y \end{bmatrix}$.

Thus, if λ is an eigenvalue of Graph G,

$$\Rightarrow$$

Given, for a graph G, for each eigenvalue $\lambda \neq 0$ there is another eigenvalue $\lambda' = -\lambda$.

We take k such that it is an arbitrary odd integer. We know that, if A has eigenvalues $\lambda_1, \ldots, \lambda_n$, then A^k has eigenvalues $\lambda_1^k, \ldots, \lambda_n^k$.

$$\text{tr}(A^k) = \sum_{i=1}^{n} \lambda_i^k = 0$$

For matrix A^k, each element $(A^k)_{ij}$, is the number of walks from i to j of length exactly k. Thus, if there is an odd cycle of length k, then $(A^k)_{ij} > 0$ so $\text{tr}(A^k) > 0$. This contradicts our previous statement that $\text{tr}(A^k) = 0$. Thus, our assumption is incorrect and there are no odd cycles of length k. Since k is chosen arbitrarily, this will hold for all odd integers. Thus, there are no odd cycles in G. Hence, by König's Theorem we can state that G is bipartite.

Eigenvalues of A(G) can be used to make more comments on the topological structure of a graph:

$\lambda_1(G) = -\lambda_n(G) \Rightarrow G$ is bipartite [Stronger statement than our proof]
$\lambda_2(G) = 0 \Rightarrow G$ is a complete multi-partite graph
$\lambda_2(G) = -1 \Rightarrow G$ is a complete graph

3.2 Algebraic Connectivity

Algebraic Connectivity of a graph allows us to quantitatively analyse how connected the vertices of the graph are based on the eigenvalues of the graph's Adjacency matrix or Laplacian Matrix. In the following section, we will use the Laplacian Matrices of simple undirected graphs in order to determine:

- The number of connected components in an undirected graph G.
- The measure of (algebraic) connectivity of a graph using μ_2 (second smallest eigenvalue) of the Laplacian Matrix of Graph G.
- Number of spanning trees that can be formed by some subset of the edges of the graph.

Greater the algebraic connectivity of the graph, lower the chances of the graph to get disconnected on the removal of vertices.

Let $\mu_1, \mu_2, \cdot \cdot \cdot \mu_{n-1}, \mu_n$ be the n eigenvalues of L(G) in ascending order. $\mu_i \geq 0$ holds for all i= 1, 2, 3, … n for any Laplacian Matrix.

$\mu_1 = 0$ always holds. If all other eigenvalues $\mu_n, \cdot \cdot \cdot, \mu_3, \mu_2$ are non-zero, the graph is connected, i.e., only one connected component exists.

The number of eigenvalues of Laplacian of a graph that are zero is equal to the number of connected components in a graph. If the vertices of a graph can be partitioned into k disjoint sets where each set of vertices is connected but there exist no edges between the vertices in the two different sets, the graph is said to have k connected components. In this case,

$$\mu_1 = \mu_2 = \mu_3 = ... = \mu_k = 0$$

Thus, the algebraic multiplicity of the eigenvalue 0 of the Laplacian Matrix L(G) gives us the number of connected components in the graph G.

3.2.1 Fielder Eigenvalue $\propto(G)$

The second smallest eigenvalue of the Laplacian Matrix known as the Fielder Eigenvalue, is a measure of the Connectivity of the Graph, called Algebraic Connectivity.

The maximum algebraic connectivity of the graph with n vertices is n.

$$0 \leq \propto(G) \leq n$$

We can set a tighter lower-bound on $\propto(G)$ as follows, where, D is diameter of the graph G, defined as the length of the shortest path between the most distanced nodes.

$$\propto(G) > \frac{4}{nD}$$

3.2.2 Spanning Trees using Laplacian Matrix

Interestingly, the product of all eigenvalues of the Laplacian starting from the second eigenvalue of a connected graph gives us the number of Spanning Trees that can be constructed using the graph.

If $\mu_1 = 0 < \mu_2 \leq \mu_3 \leq \cdots \leq \mu_n$ are the n eigenvalues of the Laplacian matrix, the number of spanning trees of the graph are given by,

$$\frac{1}{n} \prod_{i=2}^{n} \mu_i$$

This is derived from Kirchhoff's Matrix Tree Theorem which states that the number of spanning trees can be computed in polynomial time from the determinant of a submatrix of the Laplacian matrix of the graph; specifically, the number is equal to any cofactor of the Laplacian matrix.

3.2.3 Fielder Vector

The vector associated with μ_2 is known as Fielder Vector. It plays an important role in partitioning vertices of a given graph into disconnected components. Let us consider the following graph:
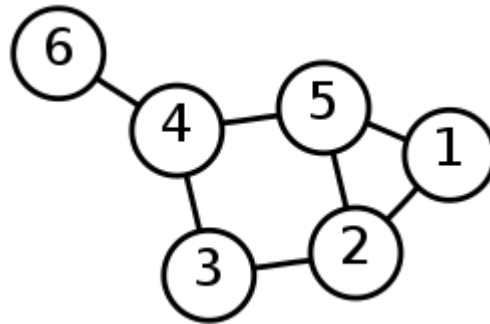
The Fielder Vector for the graph comes out to be
$$\begin{bmatrix} 0.415 \\ 0.309 \\ 0.069 \\ -0.221 \\ 0.221 \\ -0.794 \end{bmatrix}$$

The negative values at the 4th and 6th index are associated with the vertex 6, which is poorly connected and the neighbouring articulation vertex 4. Vertex 3 also has a low positive value in the Fielder Vector, allowing us to move it to the partition, creating two better connected graphs by partitioning it into 2 components {1, 2, 5} and {3, 4, 6}. The probabilities of the respective data points being assigned to the sign-based partition may be calculated from the squared values of the Fiedler vector's components, which total to one because the vector is normalised.
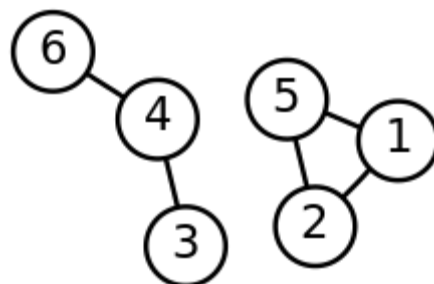
3.3 Validation and Implementation: Applying Connectivity Properties on Real Graphs

3.3.1 Analysing Connectivity Properties on a Graph

We start off with a disconnected graph and compute the adjacency matrix to the compute the Laplacian Matrix.
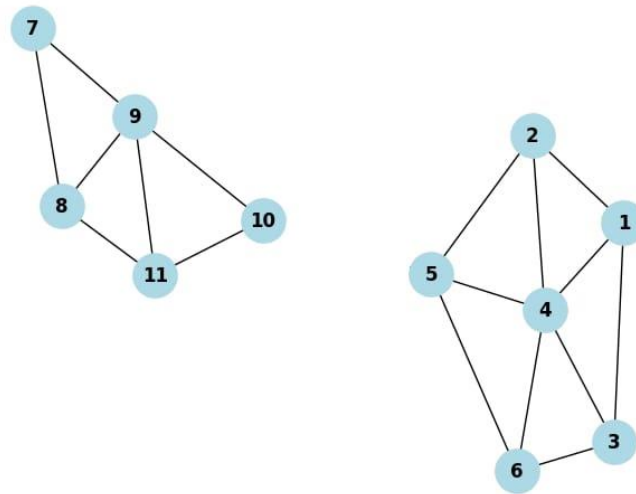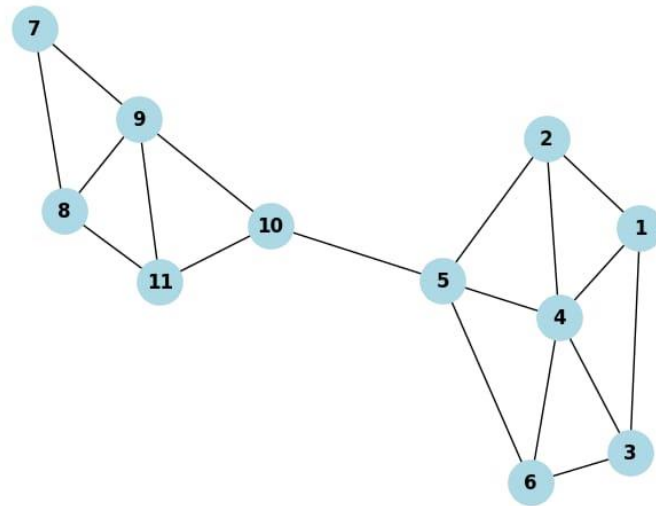

$$A=\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Now, computing the Laplacian given by $L(G) = D(G) - A(G)$,

$$L=\begin{bmatrix} 3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 3 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 5 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & 3 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 4 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 3 \end{bmatrix}$$

On computing the eigenvalues of D, we get 0, 2.3819, 4.61803, 6, 4.61803, 2.381966, 0, 1.58578, 3, 5, 4.414213; we observe that $\mu_2 = 0$, showing us that the graph is disconnected as one can observe. Obviously, no spanning trees are formed. 2 connected components are present.

We now add an edge between nodes 5 and 10 to form a weak link between the two disconnected components. Now, we re-calculate our Laplacian Matrix L' = L(G') for this graph.

$$L'=\begin{bmatrix} 3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 3 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 5 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & -1 & -1 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 4 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 3 \end{bmatrix}$$

Computing the eigenvalues, we have 0, 0.21442, 6.11044921, 1.81023809, 2.38196601, 2.65511952, 3.36802291, 5.46086479, 4.94019423, 4.61803399 and 4.44068327. As one would expect, μ_2 jumps to a positive value of 0.2114! 2541 spanning trees can be formed. Now, let us connect 4 to 9, and observe the increase in the Fielder value.
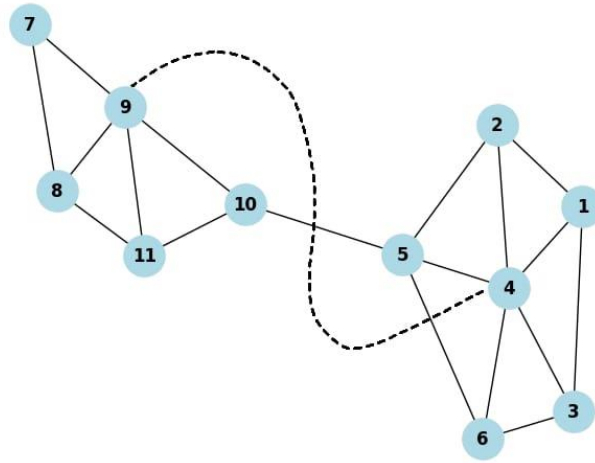
The Laplacian Matrix for the following graph is given by L" = L(G"). The Eigenvalues come out to be 7.43545, 0, 4.9891276, 1.81781461, 2.38196601 2.66158387, 3.38722078, 4.61803399, 4.44159306, 5.46416867 and 5.29326182. 7810 spanning trees can be formed.

$$
L" = \begin{bmatrix}
3 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 3 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 3 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & -1 & 6 & -1 & -1 & 0 & 0 & -1 & 0 & 0 \\
0 & -1 & 0 & -1 & 4 & -1 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & -1 & -1 & -1 & 3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & -1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 & 0 & -1 \\
0 & 0 & 0 & -1 & 0 & 0 & -1 & -1 & 5 & -1 & -1 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 3 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 3
\end{bmatrix}
$$

Since 3 and 8 were better connected nodes, this leads to a much greater magnitude of the Fielder value, $\mu_2 = 0.4989$.

3.3 Code Implementation: Computing Connectivity Properties

Constructing the Adjacency Matrix using a list of the graph's edges:

```python
edges = []
file_name="graphdata.txt"
with open(file_name, "r") as file:
    num_vertices = int(file.readline().strip())
    num_edges = int(file.readline().strip())
    edges = [tuple(map(int, line.strip().split())) for line in file]
adjacency_matrix = [[0] * num_vertices for _ in range(num_vertices)]
for edge in edges:
    vertex1, vertex2 = edge
    adjacency_matrix[vertex1-1][vertex2-1] = 1
    adjacency_matrix[vertex2-1][vertex1-1] = 1
print("Adjacency Matrix:")
for row in adjacency_matrix:
    print(row)
```

Visualizing the Graph corresponding to the Adjacency Matrix:

```python
import numpy as np
import networkx as nx
import matplotlib.pyplot as plt
graph = nx.Graph(np.array(adjacency_matrix))
graph = nx.relabel_nodes(graph, {i: i + 1 for i in graph.nodes()})
nx.draw(graph, with_labels=True, node_color='lightblue', node_size=800,font_weight='bold')
plt.show()
```

Computing the Laplacian Matrix using the Adjacency Matrix:

```python
degree_matrix = np.zeros((num_vertices, num_vertices))
for i in range(num_vertices):
    degree_matrix[i][i] = sum(adjacency_matrix[i])
laplacian_matrix = degree_matrix - np.array(adjacency_matrix)
print("Laplacian Matrix:")
for row in laplacian_matrix:
    print(row)
```

Calculating the Eigenvalues and Eigenvectors corresponding to the Laplacian:

```python
eigenvalues, eigenvectors = np.linalg.eig(laplacian_matrix)
print(f"Eigenvalues: {eigenvalues}")
print()
for i in range(len(eigenvectors)):
    print(f"Eigenvalue {i+1}: {eigenvectors[:, i]}")
    print()
```

Computing the Algebraic Multiplicity and corresponding Fielder Vector of a Graph:

```python
eigenvalues_sorted = np.sort(eigenvalues)
eigenvectors_sorted = eigenvectors[:, eigenvalues.argsort()]
alg_multiplicity = eigenvalues_sorted[1]
fielder_vector = eigenvectors_sorted[:, 1]
print("Algebraic Connectivity:", alg_multiplicity)
print("Fielder Vector:")
print(fielder_vector)
```

Calculating the number of Spanning Trees in Graph using Eigenvalues and comparing the result obtained using Kirchhoff's Method:

```python
eigenvalues_x = eigenvalues_sorted[1:]
product = np.prod(eigenvalues_x)/num_vertices
print("Number of Spanning Trees, calculated using eigenvalues:", round(product))

from scipy.linalg import det
spanning_trees_count = det(laplacian_matrix[1:, 1:])
print("Number of Spanning Trees, calculated using Kirchoff's Theorem:",round(spanning_trees_count))
```

Calculating the number of connected components in a Graph using the Algebraic Multiplicity of the zero eigenvalue, and then printing the corresponding components using Graph Traversal Algorithms:

```python
tolerance = 1e-12
zero_eigenvalues_count = np.sum(np.abs(eigenvalues) < tolerance)
print("Number of Connected Components in the graph are:", zero_eigenvalues_count)

def dfs(graph, start, visited, component):
    visited.add(start)
    component.append(start)
    for neighbor in graph.neighbors(start):
        if neighbor not in visited:
            dfs(graph, neighbor, visited, component)
connected_components = []
visited = set()
for node in graph.nodes():
    if node not in visited:
        component = []
        dfs(graph, node, visited, component)
        connected_components.append(component)
print("Connected Components:")
for i, component in enumerate(connected_components):
    print(f"Component {i + 1}: {component}")
```

3.4 Applications

3.4.1 Network Robustness

Algebraic connectivity is vital in assessing the robustness of networks in various domains, including power grids and transportation networks. Possessing the ability to quantify the degree of connectivity within a graph, it indicates if the system possesses the ability to withstand failures or disruptions.

In terms of network robustness, a higher algebraic connectivity value signifies a more resilient network. If a node or link fails, a graph with high algebraic connectivity is more likely to maintain its overall connectivity compared to a graph with lower algebraic connectivity. The presence of strong interconnections ensures alternative paths for information flow, mitigating the impact of failures and enhancing the network's ability to function effectively even under adverse conditions.

Power grids rely on the efficient transmission of electricity across a vast network. The algebraic connectivity allows assessment of the grid's robustness to failures from deliberate attacks or wiring issues. By analysing the graph's eigenvalues and eigenvectors, one can identify critical nodes or links that, on disruption, would have a significant impact on the connectivity of the grid.

Communication networks, such as the internet or telecommunications networks, are highly interconnected systems that rely on reliable information exchange. The algebraic connectivity enables the evaluation of their ability to recover from failures or attacks. Network operators, using linear algebra, can identify critical communication links and design redundancy mechanisms to ensure continuous data transmission.

3.3.2 Community Detection

The task of identifying cohesive groups or communities within a network is known as Community. Algebraic connectivity is used as to identify communities. A higher algebraic connectivity value suggests stronger intra-community connections and weaker inter-community connections, allowing detecting and classifying community structures.

3.3.3 Graph Partitioning

Dividing a graph into subsets of nodes with specific properties is called Graph Partitioning. Algebraic connectivity is used as a measure for graph partitioning algorithms. Maximizing the algebraic connectivity during partitioning helps create partitions with better interconnectivity and facilitates efficient distributed computing, load balancing, and resource allocation in parallel computing environments.

4. Spectral Clustering

Spectral clustering is a technique which is used to identify the clustering of nodes based on the edges connecting them. It uses information from the eigenvalues (spectrum) of special matrices built from the graph or the data set. This allows for more varied form of clusters compared to other techniques and the technique is effective for data of different shapes and size.

For example, K-means algorithm uses compactness approach. This helps in grouping the data based on the distance between the data points. But this also limits the accuracy or the consistency of the clusters. K-means algorithm generally assumes that the clusters are spherical or round i.e. within k-radius from the cluster centroid. In K means, many iterations are required to determine the cluster centroid.

 In spectral, the clusters do not follow a fixed shape or pattern. It helps us overcome two major problems in clustering: the shape of the cluster and the other is determining the cluster centroid.
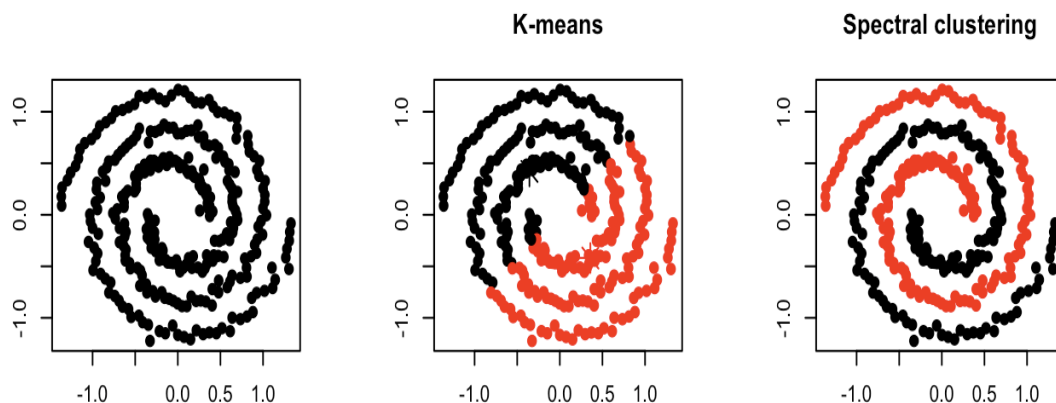
Graph Laplacians

The main tools used for spectral clustering are graph Laplacian matrices.
The unnormalized graph Laplacian matrix is defined as

$$L = D - W$$

Properties of L

- For every vector $f \in R^n$ we have

$$f'Lf = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2.$$

- L is symmetric and positive semi-definite.
- The smallest eigenvalue of L is 0, the corresponding eigenvector is the constant one vector 1.
- L has n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

The normalized graph Laplacians

$$L_{\text{sym}} := D^{-1/2}LD^{-1/2} = I - D^{-1/2}WD^{-1/2}$$
$$L_{\text{rw}} := D^{-1}L = I - D^{-1}W.$$

the first matrix by L_{sym} as it is a symmetric matrix, and the second one by L_{rw} as it is closely related to a random walk.

Properties of L_{sym} and L_{rw}

- For every $f \in R^n$

$$f'L_{sym}f = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2.$$

- λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ is an eigenvalue of L_{sym} with eigenvector $D^{1/2}u$.
- λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ and u solve the generalized eigenproblem $Lu = \lambda Du$.
- 0 is an eigenvalue of L_{rw} with the constant one vector 1 as eigenvector. 0 is an eigenvalue of L_{sym} with eigenvector $D^{1/2}u$.
- L_{sym} and L_{rw} are positive semi-definite and have n non-negative real-valued eigenvalue $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

Eigenvalues of Graph Laplacian
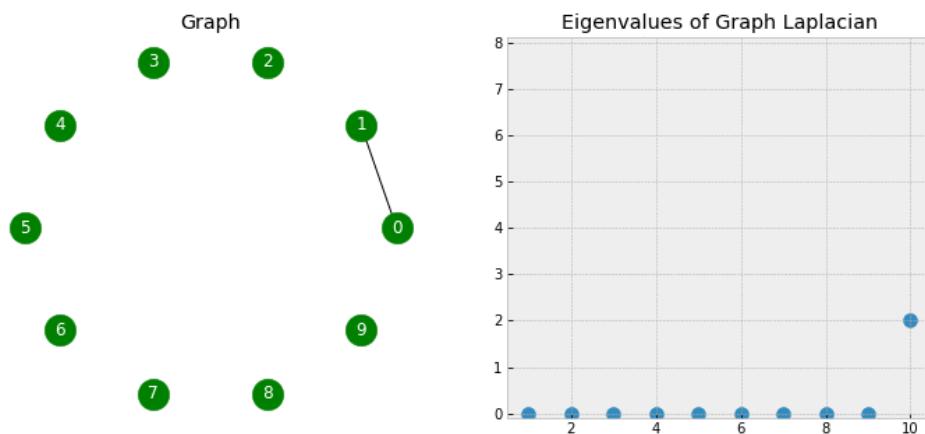
When the graph is completely disconnected, all ten of our eigenvalues are 0. But as the edges increase so do the eigen values. So eigenvalues correspond to the number of connections in the graph.

Spectral clustering algorithms

Unnormalized spectral clustering

Input: Similarity matrix $S \in R^{n \times n}$, number k of clusters to construct

- Construct a similarity graph. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L.
- Compute the first k eigenvectors $u_1,...,u_k$ of L.
- Let $U \in R^{n \times k}$ be the matrix containing the vectors $u_1,...,u_k$ as columns.
- For i = 1,...,n, let $y_i \in R^k$ be the vector corresponding to the i-th row of U.
- Cluster the points $(y_i)_{i=1,...,n}$ in k with the k-means algorithm into clusters $C_1,...,C_k$.

Output: Clusters $A_1,...,A_k$ with $A_i = \{j| y_j \in C_i\}$.

Normalized spectral clustering according to Shi and Malik

Input: Similarity matrix $S \in R^{n \times n}$, number k of clusters to construct

- Construct a similarity graph. Let W be its weighted adjacency matrix.

- Compute the unnormalized Laplacian L.
- Compute the first k generalized eigenvectors $u_1,...,u_k$ of the generalized eigenproblem $Lu = \lambda Du$.
- For i = 1,...,n, let $y_i \in R^k$ be the vector corresponding to the i-th row of U.
- Cluster the points $(y_i)_{i=1,...,n}$ in k with the k-means algorithm into clusters $C_1,...,C_k$.

Output: Clusters $A_1,...,A_k$ with $A_i = \{j| y_j \in C_i\}$.

Normalized spectral clustering according to Ng, Jordan, and Weiss

Input: Similarity matrix $S \in R^{n \times n}$, number k of clusters to construct.

- Construct a similarity graph. Let W

be its weighted adjacency matrix.
- Compute the normalized Laplacian Lsym.
- Compute the first k eigenvectors $u_1,...,u_k$ of Lsym.
- Let $U \in R^{n \times k}$ be the matrix containing the vectors $u1,...,uk$ as columns.
- Form the matrix $T \in R^{n \times k}$ from U by normalizing the rows to norm 1, that is set $t_{ij} = u_{ij} / \sum_k u_{ik}^2{}^{1/2}$
- For $i = 1,...,n$, let $y_i \in R^k$ be the vector corresponding to the i-th row of T.
- Cluster the points $(y_i)_i = 1,...,n$ with the k-means algorithm into clusters $C_1,...,C_k$.

Output: Clusters $A_1,...,A_k$ with $A_i = \{j|\ y_j \in C_i\}$.

5. Eigenvector Centrality

A natural extension of a degree centrality is **eigenvector centrality.** In-degree centrality awards one point for every link a node receives.

But not all vertices are equivalent: some are more relevant than others.

The Eigenvector centrality thesis reads:

> A node is important if it is linked to by other important nodes.

But a node receiving many links may not have a high eigenvector centrality as all its linkers might have low eigen centrality or a node with high eigenvector centrality might not be highly linked as its linkers might have high eigenvector centrality.

Using the adjacency matrix to find eigenvector centrality.

For a given graph $G := (V, E)$ with $|V|$ vertices let $A = (a_{v,t})$ be the adjacency matrix, i.e $a_{v,t} = 1$ if vertex v is linked to vertex t, and $a_{v,t} = 0$ otherwise. The relative centrality score, x_v, of vertex v can be defined as:

$$x_v = \frac{1}{\lambda} \sum_{t \in M(v)} x_t = \frac{1}{\lambda} \sum_{t \in V} a_{v,t} x_t$$

Where $M(v)$ is the set of neighbours of v and λ is a constant. With a small rearrangement this can be written in vector notation as the eigenvector equation

$$Ax = \lambda x$$

The vth component of related eigenvector then gives the relative centrality score of vertex v in the network. The eigenvector is only defined up to a common factor, so only the ratios of the centralities of the vertices are well defined. To define an absolute score, one must normalise the eigenvector e.g. such that the sum over all vertices is 1 or the total number of vertices n.

Power iteration is one of many eigenvalue algorithms that may be used to find this dominant eigenvector.

Applications

- Eigenvector centrality is the unique measure satisfying certain axioms for a ranking system.
- In neuroscience, the eigenvector centrality of a neuron in a model neural network has been found to correlate with its relative firing rate.
- Eigenvector centrality and related concepts have been used to model opinion influence in sociology and economics, as in the DeGroot learning model.
- In a study using data from the Philippines, researchers showed how political candidates' families had disproportionately high eigenvector centrality in local intermarriage networks.[13]
- Eigenvector centrality has been extensively applied to study economic outcomes, including cooperation in social networks.[14] In economic public goods problems, a person's eigenvector centrality can be interpreted as how much that person's preferences influence an efficient social outcome.

6. Resilience

The resilience is property of graph structure refers to its ability to maintain its essential properties and functionality even when subjected to various forms of stress or attacks (by stress or attacks we mean removing certain edges or nodes of the graph). It measures graph's ability to withstand and recover from disruption, such as removal of nodes or edges, changes in connectivity patterns, or targeted attacks on specific components. Resilience is an important characteristic of graph structure and understanding it helps in designing robust systems that can continue to function effectively even in the face of challenges.
In this project we aim to analyse the relationship between eigen values and resilience of a graph.

6.1 Different Aspects of Resilience

A graph structure has following different aspects of resilience –

1. Connectivity – This property of graphs determines how easily information can flow between its nodes. A resilient graph maintains high level of connectivity even when certain edges are removed.

2. Redundancy – It refers to the existence of multiple paths within a graph. A resilient graph often incorporates redundancy by having multiple edges between the nodes that can take over the functionality of failed nodes.

3. Robustness – Robustness is a measure of the ability of the graph to tolerate random failures without significant degradation in its performance. A resilient graph is robust in nature and can withstand failures without significant loss of connectivity or functionality.

4. Attack Resistance - This includes identifying key nodes or edges that, if removed, would cause the most disruption to the graph's connectivity or efficiency. Resilient graphs are designed to minimize the impact of such targeted attacks and maintain their functionality even under deliberate efforts to disrupt them.

5. Recovery - Resilience is not just about withstanding disruptions but also about recovering from them efficiently. A resilient graph can quickly adapt and restore its connectivity or functionality after a disruption has occurred.

6.2 How the Eigen Values of a Graph play a role in determining Resilience of a Graph ??

Eigen Values play a crucial role in understanding and analysing the resilience of graph structures. They provide valuable insights into the graph's connectivity, structure, and resilience.

6.2.1 Algebraic Connectivity

As discussed earlier, the algebraic connectivity of a graph is determined by the second smallest eigenvalue of the Laplacian matrix, called the Fielder Value. It is a fundamental indicator of graph resilience.
The larger Fielder Value, the more connected the network, and thus more opportunities for a graph to recover efficiently after facing attacks. The Fielder value can yield insight into the network's structure, and thereby serve as the basis for introducing additional resilience in the graph's structure by judiciously introducing extra edges (or, conversely, give an adversary a tool to identify weak points of a network where an attack can have a maximal impact).

Fielder value is always non-negative, and its value is zero if and only if the graph is disconnected (in this case the number of zero eigenvalues of L equals the number of connected components in a graph). For a strongly connected graph the Fielder value, is always larger than zero.

The Fielder Value is denoted by μ_2.

If an attack kills the link (edge of the graph) in between two nodes, thereby producing a new edge set E_1, then the new Fielder value satisfies,

$$\mu_2 (V, E_1) \leq \mu_2 (V, E), \quad \text{where } E_1 \text{ is a subset of E.}$$

The Fielder value's upper bound is limited by the minimum degree of nodes and the total number of nodes that exist in the network.

$$\mu_2 (V, E) \leq \min d_v (|V| / |V| -1)$$

From this, we can observe that the Fielder Values can become large when adding an edge to the graph, and thus the graph can become more resilient by increasing Fielder Value.

Since we are interested in the vulnerability of a graph when a node is removed from a graph, we now introduce a modified notion of the Fiedler value, which corresponds to role of a particular node in the network's connectivity as measured by the impact associated with removing all that node's edges. Specifically, we propose a new measure of connectivity, which we term a node's Fiedler value :

" For a graph G(V, E), the node Fiedler value associated with node j corresponds to the Fiedler value $\mu_2 (V, E_j)$, where E_j corresponds to a revised set of edges for G where all edges containing node j have been removed from E. "

The most connectivity influential node in the graph will be the one whose removal would weaken the graphs connectivity the most. To find this node we find the fielder values after removing each node and discover the node which effects the fielder value and therefore impact its resilience the most.

Procedure to find the node that has most harmful impact on graphs resilience :
1. Remove each node, i, in a graph G
2. Calculate the Fielder value μ_i, i = 1,…,n for the remaining Graph where n is the number of nodes in the graph.
3. The node with most significant impact on the connectivity and hence the resilience will be

$$C = \arg_i \max \mu_i$$

For example, for a 5-node graph, the fielder values for the star topology is μ = 1, compared to μ = 0.382 for a line topology.
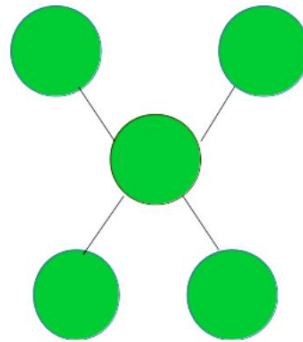
5 Node Star Topology Graph

If in the star topology graph the central node is one then, its Laplacian matrix is given by –

$$L = \begin{vmatrix} 4 & -1 & -1 & -1 & -1 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 1 \end{vmatrix}$$

If we remove central node (i.e., Node 1) in this graph that connects all the other nodes the Fielder value will become zero (in fact, we will have a totally disconnected network). This will mean that the central node will has more influence on connectivity than removing the leaf node.

From the above discussion we can conclude that the Algebraic Connectivity (i.e., the Fielder Value of a graph) play a major role in determining the resilience of a graph towards external attacks and in determining the nodes which are most sensitive towards the attacks. This information helps in creating graphs which can withstand node or edge removals without becoming disconnected.

6.2.2 Spectral Gap

The spectral gap of a graph is defined as the absolute difference between the largest and second largest eigen values of the Laplacian matrix of the graph. The spectral gap is a measure that captures the spread or distribution of eigenvalues in a graph's Laplacian matrix.

A larger spectral gap indicates a more distinct eigenvalue distribution, implying a more resilient graph with better connectivity and robustness.

Let's consider a case of three 5 nodes graphs which different topologies.

1. Line Graph - A line graph is a simple graph where the nodes are arranged in a linear sequence, and each node is connected to its adjacent nodes. The line graph can be thought of as a straight line with nodes placed along it.
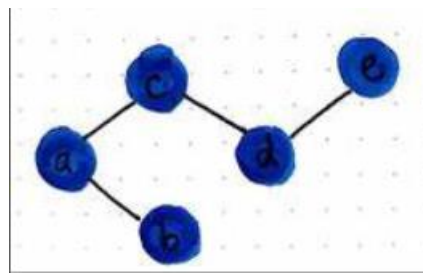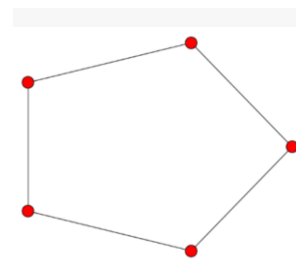

5 node line graph

2. Cyclic Graph - A cyclic graph, also known as a cycle or circular graph, forms a closed loop or cycle where each vertex is connected to two other vertices

3. Complete Graph - A complete graph is a graph where every pair of distinct nodes is connected by an edge. In other words, in a complete graph, there is an edge between every pair of nodes.
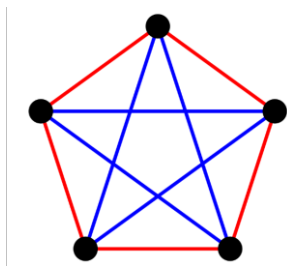
5 node complete graph

5 node cyclic graph

The Laplacian and Adjacency Matrix for a Single Chain or a Line graph is:

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Adjacency Matrix for Line Graph

$$\begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

Laplacian Matrix for Line Graph

The spectral gap for a line graph is given as $\mu_1 - \mu_2 = 3.618 - 2.618 = 1$.

The Laplacian and Adjacency Matrix for a Cyclic Graph is :

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Adjacency Matrix for Cyclic Graph

$$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{pmatrix}$$

Laplacian Matrix for a Cyclic Graph

The spectral gap for a line graph is given as $\mu_1 - \mu_2 = 3.61 - 1.38 = 2.23$.

The Laplacian and Adjacency Matrix for a Complete Graph is :

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Adjacency Matrix for Complete Graph

$$\begin{pmatrix} 4 & -1 & -1 & -1 & -1 \\ -1 & 4 & -1 & -1 & -1 \\ -1 & -1 & 4 & -1 & -1 \\ -1 & -1 & -1 & 4 & -1 \\ -1 & -1 & -1 & -1 & 4 \end{pmatrix}$$

Laplacian Matrix for Complete Graph

The spectral gap for a line graph is given as $\mu_1 - \mu_2 = 5 - 0 = 5$.

Analysing the connectivity of the Graphs –

1. A line graph is a connected graph since every node is connected to its adjacent nodes. However, removing any node from the line graph breaks the graph into two disconnected components. It possesses a small spectral gap as calculated above (=1), indicating its lower resilience to attacks.

2. The cyclic graph has higher spectral gap in comparison to a line graph, showing it is more resilient to outside attacks as compared to a line graph. If we remove a single edge from this graph, it will remain connected. However, removing 2 or more edges will break the graph into disconnected components.

3. A complete graph offers a direct connection between every pair of vertices. A complete graph ensures that there are no isolated vertices, and it maximizes the number of edges in the graph, providing the highest level of connectivity. The high spectral gap of Complete Graphs indicates their high connectivity and resilience to outside attacks. In this case only on removing 4 or more edges may disconnect some vertices of the graph which proves its high resilience.

6.3 Applications

- Understanding the resilience of graphs helps in designing and analysing robust networks, such as communication networks, power grids, transportation systems, or social networks. By assessing the resilience of these networks, one can identify critical components or vulnerabilities that may lead to failures or disruptions.

- Graph resilience is crucial in studying the spread of diseases and designing effective strategies for containment. By analysing the resilience of contact networks, researchers can identify key individuals or regions that, if targeted, can effectively limit the spread of a disease or prevent an epidemic outbreak.

- Resilience analysis is valuable in studying the stability and performance of computer networks, including the Internet. It helps identify potential bottlenecks, vulnerabilities to cyberattacks, and methods to enhance network reliability, fault tolerance, and load balancing.

- Resilience analysis of social and economic networks provides insights into the stability and sustainability of these systems. It helps understand how shocks, such as financial crises or social disturbances, propagate through networks and impact the overall system. This knowledge can assist policymakers in designing interventions and strategies to enhance system resilience.

7. Use case of Linear Algebra

- Linear algebra techniques are used to calculate eigenvalues and eigenvectors of the graph's adjacency matrix.
- These eigenvalues are used to calculate the spectral gap of the matrix (spectral gap is the difference between the largest and second largest eigenvalue of the adjacency matrix). This spectral gap provides insights into the resilience of the graph by using linear algebraic operations.
- The eigenvalues of the Laplacian matrix also help in calculation the Fielder Value of the graph which has many important use cases.
- Linear algebraic tools such as Laplacian matrix are be utilized to analyse the connectivity properties of the graph.
- Various Linear Algebraic applications are be used in the clustering algorithm that utilizes the eigenvalues and eigenvectors of the Laplacian Matrix.

8. Conclusion

In this project we have explored the relationship between eigenvalues and graph properties, such as connectivity, clustering, and resilience. By analysing the eigenvalues of graphs with varying characteristics, we have uncovered patterns and relationships that provide insights into the interplay between graph theory and linear algebra. The findings of this project have demonstrated that eigenvalues offer valuable information about the structure of graphs.

The spectral gap, which measures the separation between the largest and second-largest eigenvalues and the Fielder Value have been shown to be a useful indicator of graph resilience. Additionally, the project has revealed that eigenvalues provide insights into graph connectivity. Moreover, the analysis of eigenvalues has implications for clustering in graphs. The distribution of eigenvalues can reveal the presence of distinct clusters or

communities within a graph, providing a glimpse into its modular structure.

Overall, the project highlights the significance of eigenvalues as a powerful tool for analysing and understanding various graph properties, paving the way for further research in this exciting intersection of graph theory and linear algebra.

9. References

- [Applications of Graph Connectivity to Network Security - ScienceDirect](#)
- https://pegasus.uprm.edu/xryong/SelectedTalks/Spring12-Talk.pdf
- http://www.ams.org/bookstore/pspdf/cbms-115-prev.pdf
- https://people.csail.mit.edu/dsontag/courses/ml14/notes/Luxburg07_tutorial_spectral_clustering.pdf
- https://www.analyticsvidhya.com/blog/2021/05/what-why-and-how-of-spectral-clustering/
- https://towardsdatascience.com/spectral-clustering-aba2640c0d5b
- https://www.geeksforgeeks.org/eigenvector-centrality-centrality-measure/
- https://www.researchgate.net/publication/275541345_Resilience_Notions_for_Scale-free_Networks