Linear Algebra - Project

Team 64- Meet Gera   Yashas S. B. Rohan

Topic: Investigating the relationship between eigenvalues and graph properties

Types of Matrices

Related to this project

Adjacency and Affinity Matrix (A)

- The graph (or set of data points) can be represented as an Adjacency Matrix, where the row and column indices represent the nodes, and the entries represent the absence or presence of an edge between the nodes.

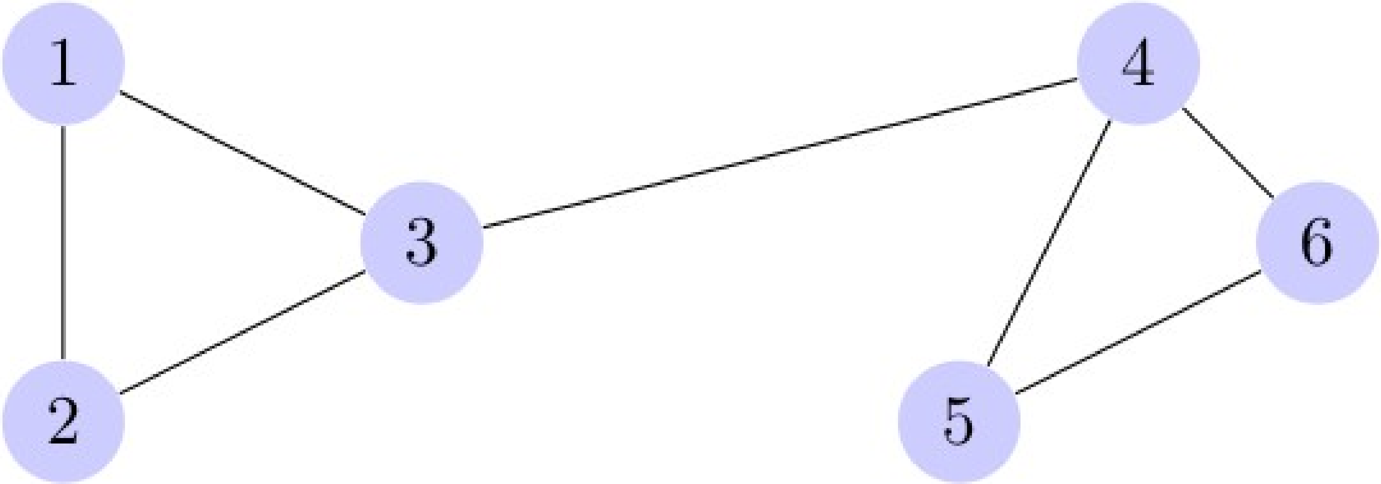
Degree Matrix (D)

- A Degree Matrix is a diagonal matrix, where the degree of a node (i.e. values) of the diagonal is given by the number of edges connected to it. We can also obtain the degree of the nodes by taking the sum of each row in the adjacency matrix.

Laplacian Matrix (L)

- Laplacian matrix is obtained by subtracting the Adjacency Matrix from the Degree Matrix (i.e.L = D–A).

Example



For the above graph

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

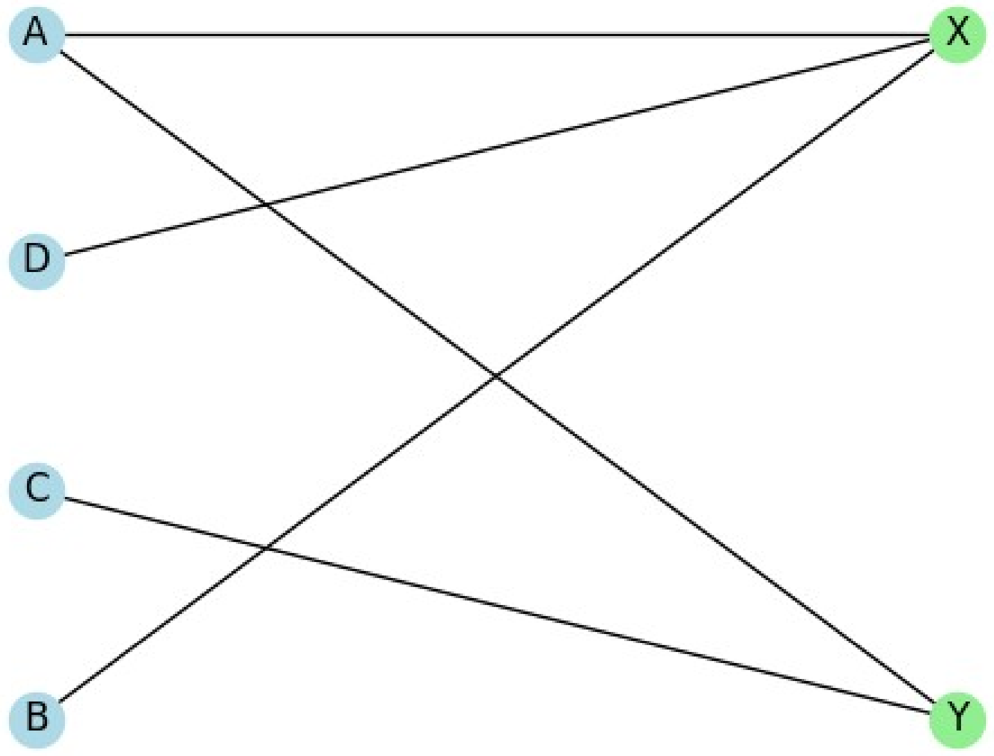
$$D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}$$

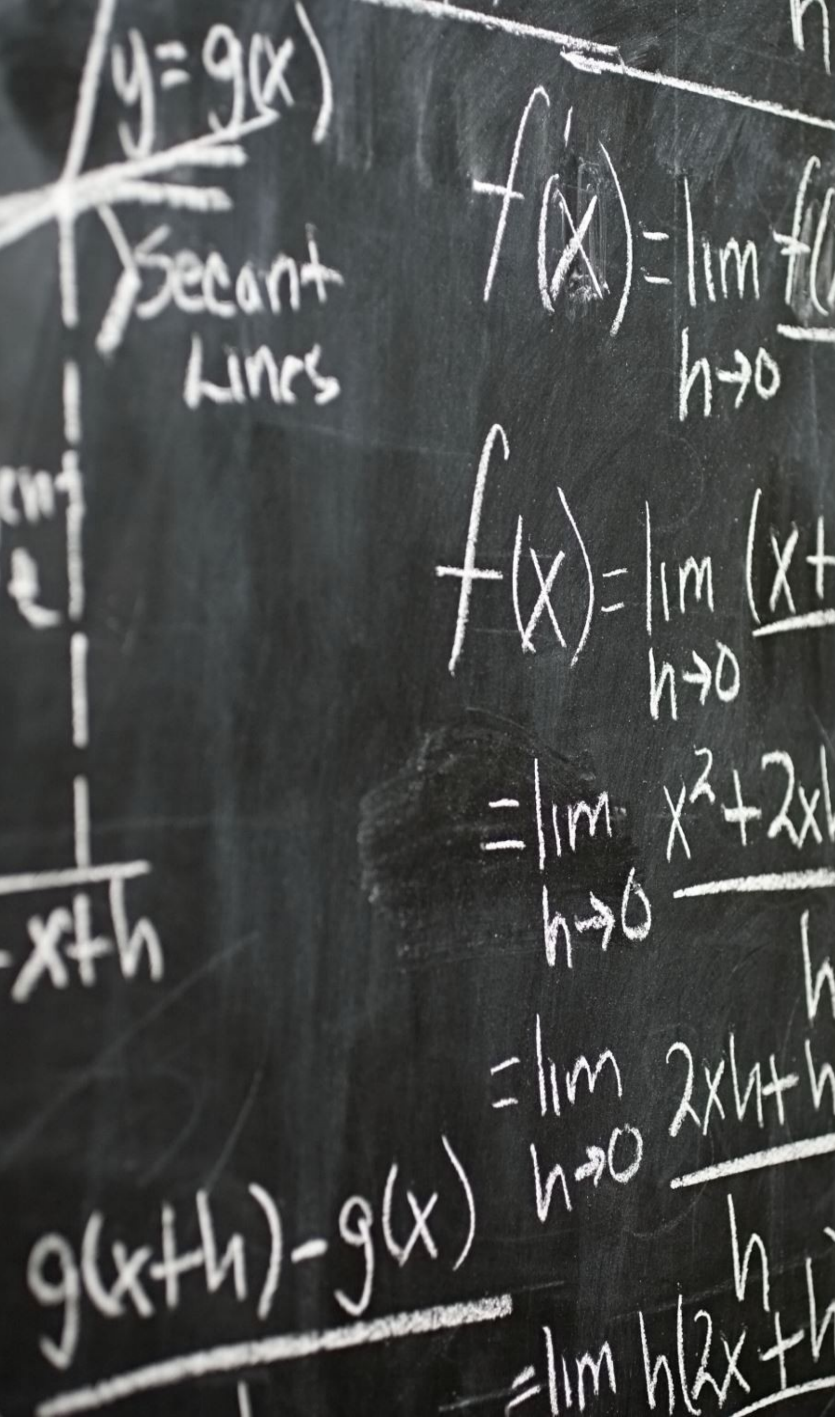
$$L = \begin{bmatrix} 3 & -1 & 0 & 0 & -1 & -1 \\ -1 & 2 & -1 & 0 & -1 & 0 \\ -1 & -1 & 3 & -1 & 0 & 0 \\ -1 & 0 & -1 & 3 & -1 & -1 \\ 0 & -1 & 0 & -1 & 2 & -1 \\ 0 & -1 & -1 & 0 & -1 & 3 \end{bmatrix}$$

Bipartite Graphs

How does maximal optimisation for favours from two parties work?

What Are Bipartite Graphs ?





Mathematical Representation.

- Graph is Collection of Nodes and Edges

Where

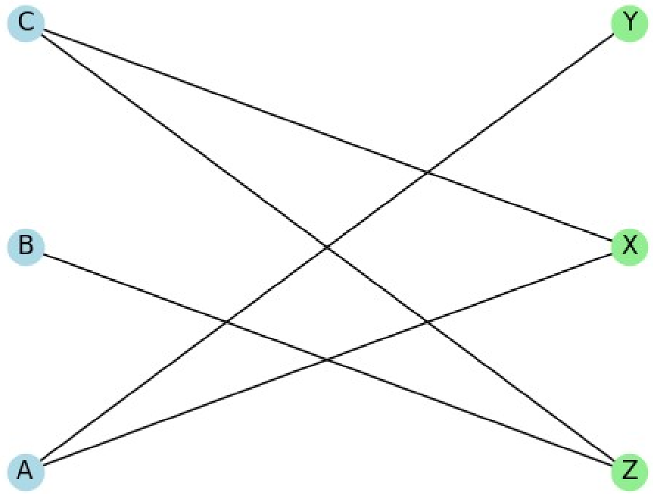
- Bipartite Graphs have two set of Nodes such that There is no edge between any two node of one set.

Mathematically :

Bipartite Graphs representation :

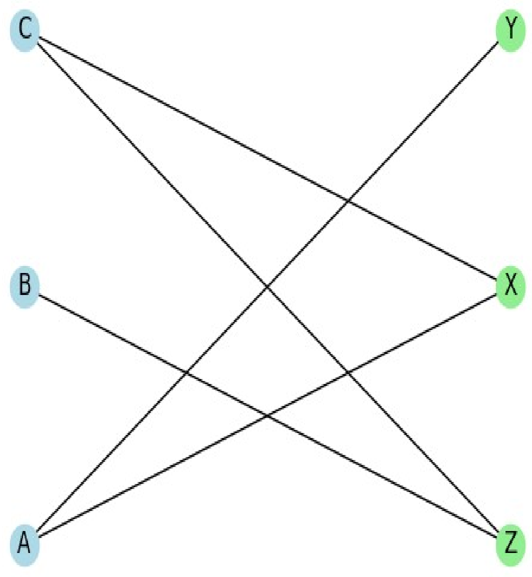
```python
import networkx as nx
import matplotlib.pyplot as plt
import numpy as np

# Create an instance of the bipartite graph
B = nx.Graph()

# Define the two sets of vertices
V1 = {'A', 'B', 'C'}
V2 = {'X', 'Y', 'Z'}

# Add the vertices to the graph
B.add_nodes_from(V1, bipartite=0)  # Set bipartite=0 for the first
set
B.add_nodes_from(V2, bipartite=1)  # Set bipartite=1 for the second
set

# Define the edges
edges = [('A', 'X'), ('A', 'Y'), ('B', 'Z'), ('C', 'X'), ('C', 'Z')]

# Add the edges to the graph
B.add_edges_from(edges)

# Separate the vertices into two sets based on the bipartite
attribute
X, Y = nx.bipartite.sets(B)

# Visualize the bipartite graph
pos = nx.bipartite_layout(B, X)

# Create a new figure for the first plot
plt.figure(1)

# Draw the vertices from each set with different colors
nx.draw_networkx_nodes(B, pos, nodelist=X, node_color='lightblue')
nx.draw_networkx_nodes(B, pos, nodelist=Y, node_color='lightgreen')

# Draw the edges
nx.draw_networkx_edges(B, pos)

# Draw labels for the vertices
nx.draw_networkx_labels(B, pos)

# Display the first plot
plt.axis('off')

adj_matrix = nx.to_numpy_matrix(B, nodelist=sorted(B.nodes()))

# Create a new figure for the second plot
plt.figure(2)

# Plot the adjacency matrix as a heatmap
plt.imshow(adj_matrix, cmap='viridis', interpolation='nearest')
plt.title('Adjacency Matrix')
```

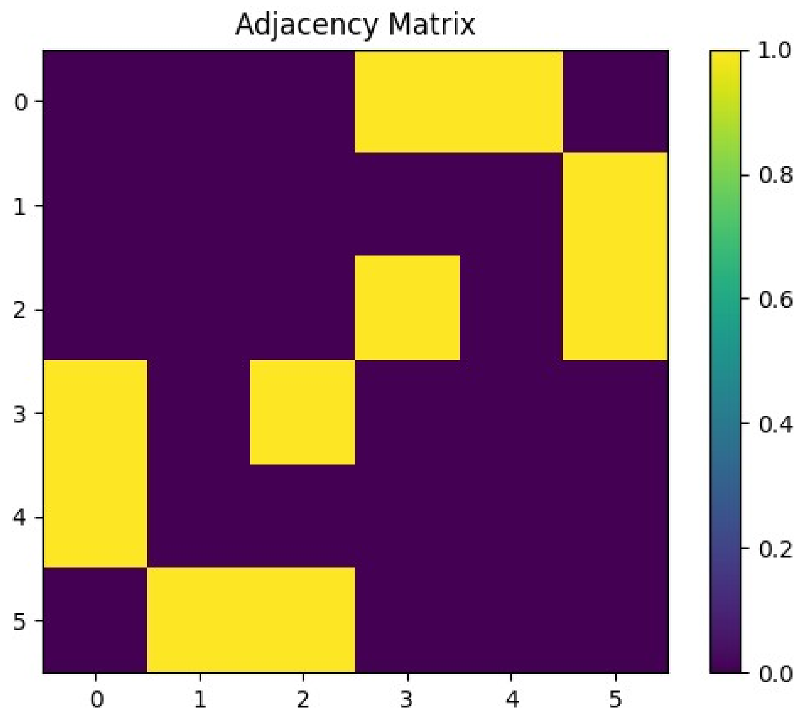
Visualisation

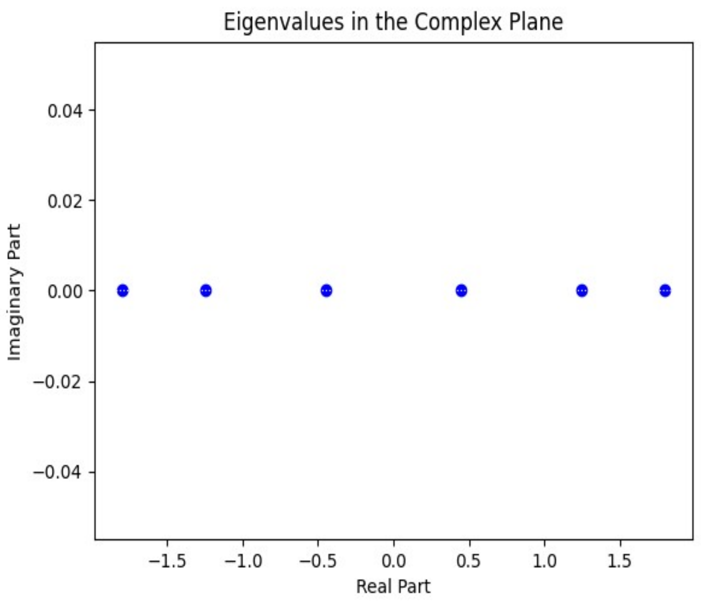


**Codes are also in submission

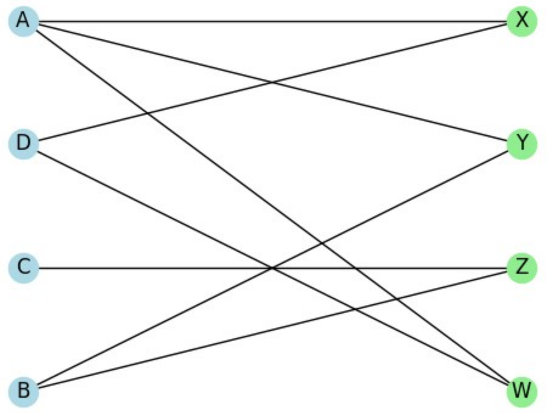
Adjoint Matrix , Graph and Eigen Plot

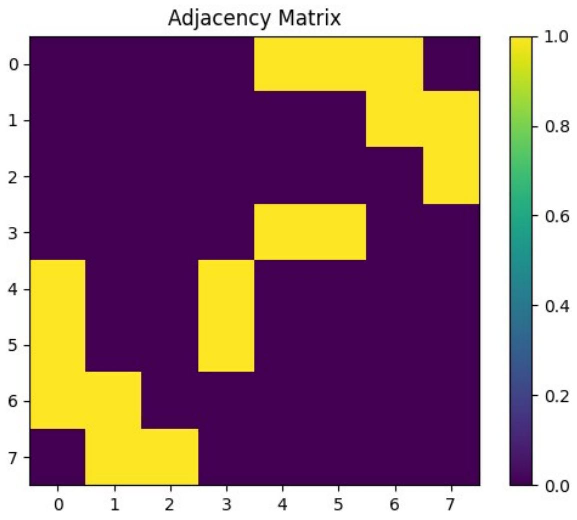


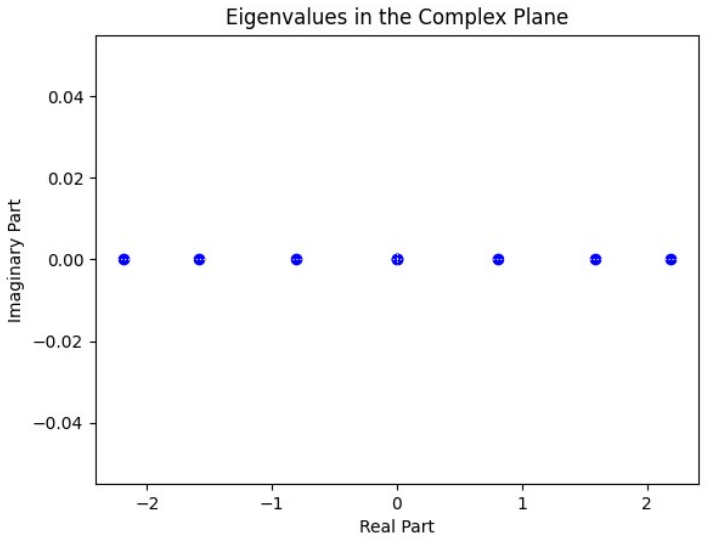




Adjoint Matrix , Graph and Eigen Plot







Applications

- Matching and Assignment Problems :

Entries from one set need to be paired with another based on certain criteria

Bipartite Graphs model such problems and device efficient algorithms to find optimal solutions.

Applications of this include Resource Allocation, Job Assignment and project Scheduling.

Biological and Social Networks

- Bipartite graphs find applications in analyzing biological networks, such as protein-protein interaction networks or gene-disease association networks.
- Similarly, bipartite graphs are utilized in social network analysis to study interactions between individuals and groups, friendship networks, and information diffusion.

Relation with Eigenvalue and Proof

Theorem 1 (Eigenvalues of Bipartite Graphs). *Let* $\mathbf{A}$ *be the adjacency matrix of a bipartite graph. By definition, a bipartite graph has its vertices divided into two disjoint sets such that all edges connect vertices from different sets. The eigenvalues of* $\mathbf{A}$ *appear in pairs of* $+a$ *and* $-a$*, where* a *is a real number.*

Proof. We will prove the statement by considering the properties of the adjacency matrix of a bipartite graph.

Step 1: Matrix A^2

Consider the matrix $\mathbf{A}^2$. The (i, j)th entry of $\mathbf{A}^2$ represents the number of paths of length 2 between vertex i and vertex j. In a bipartite graph, there are no direct edges between vertices within the same set, so all entries on the diagonal of $\mathbf{A}^2$ are zero.

Step 2: Equivalence of A^2v and Av^2

For any vector $\mathbf{v}$, we have $(\mathbf{A}^2)\mathbf{v} = \mathbf{A}(\mathbf{v}^2)$. This equation can be proven using matrix multiplication.

Step 3: Eigenvalue λ and eigenvector $\mathbf{v}$

Let λ be an eigenvalue of $\mathbf{A}$ with eigenvector $\mathbf{v}$. By definition, $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$.

Step 4: From $Av = \lambda v$ to $Av^2 = \lambda(Av)$

Multiply both sides of the equation by $\mathbf{A}$: $\mathbf{A}(\mathbf{Av}) = \mathbf{A}(\lambda\mathbf{v})$. This yields $\mathbf{A}^2\mathbf{v} = \lambda(\mathbf{Av})$.

Step 5: Substituting Step 2

By substituting Step 2, we have $\mathbf{Av}^2 = \lambda(\mathbf{Av})$. This implies that λ is an eigenvalue of $\mathbf{A}^2$ with eigenvector $\mathbf{v}^2$.

Step 6: Non-negativity of A^2

The eigenvalues of $\mathbf{A}^2$ are non-negative. This is because $\mathbf{A}^2$ is a non-negative matrix representing the number of paths of length 2 in the bipartite graph.

Step 7: Eigenvalue λ is non-negative

Since λ is an eigenvalue of $\mathbf{A}^2$, we have $\lambda \geq 0$.

Step 8: Assumption: $\lambda \neq 0$

Let's assume $\lambda \neq 0$. If $\lambda \neq 0$, then the eigenvector $\mathbf{v}^2$ is nonzero.

Step 9: From $Av = \lambda v$ to $\lambda = \frac{v^T A^T v}{v^T v}$

By the definition of the eigenvalue, we have $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$. Multiplying both sides of the equation by $\mathbf{v}^T$, we get $(\mathbf{v}^T\mathbf{A}\mathbf{v}) = \lambda(\mathbf{v}^T\mathbf{v})$.

Step 10: Symmetric property

Since $\mathbf{A}^T = \mathbf{A}$ for a bipartite graph, we have $\mathbf{A}^T\mathbf{v} = \mathbf{A}\mathbf{v}$. This implies that $(\mathbf{v}^T\mathbf{A}^T\mathbf{v}) = (\mathbf{v}^T\mathbf{A}\mathbf{v})$.

Step 11: Combining Step 9 and Step 10

Combining steps 9 and 10, we get $\lambda = \frac{v^T A^T v}{v^T v}$.

Step 12: Eigenvalue λ is real

Since $\lambda = \frac{v^T A^T v}{v^T v}$, it follows that λ is a real number.

Step 13: Conclusion: Eigenvalues appear in pairs

Therefore, we conclude that the eigenvalues of $\mathbf{A}$ appear in pairs of $+a$ and $-a$, where a is a real number.

$\square$

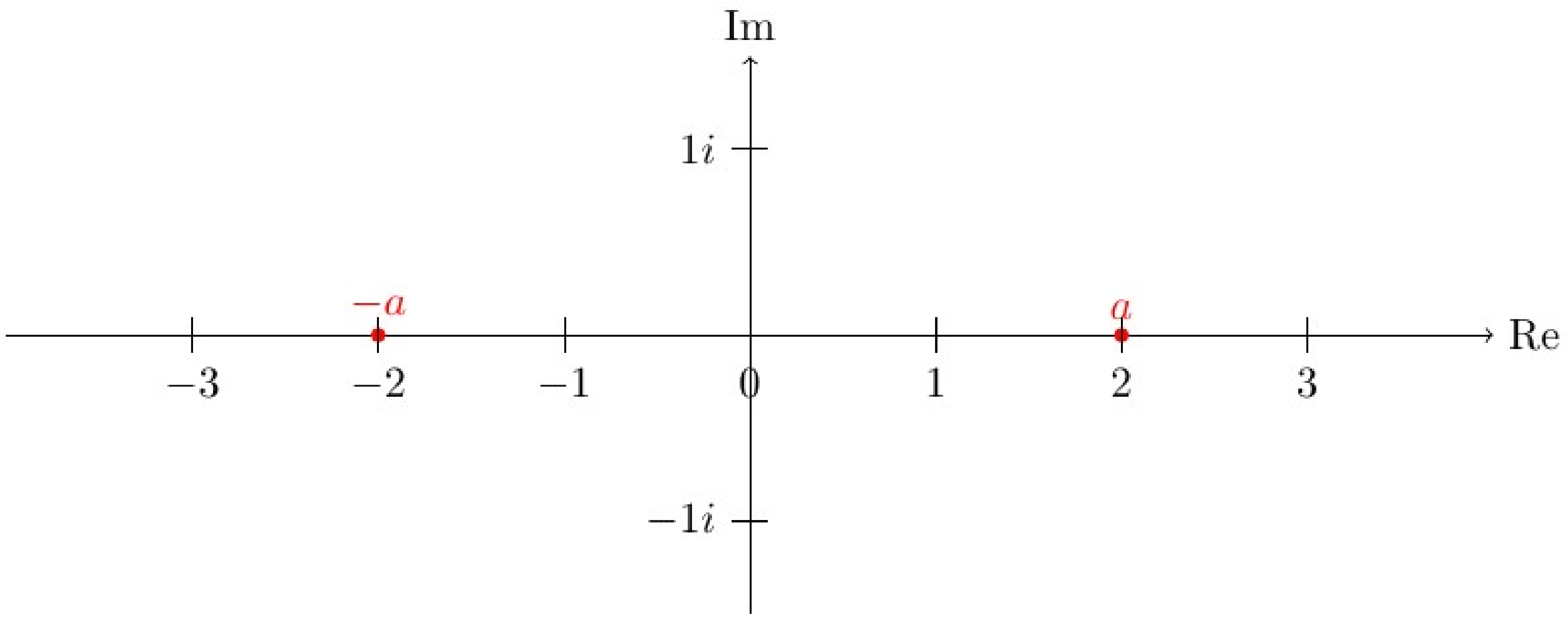


Figure 3: Eigenvalues of a Bipartite Graph

SPECTRAL CLUSTERING

What is it?

Spectral clustering is a machine learning technique that groups data points into clusters based on their spectral properties, which capture the underlying structure of the data. It utilizes the eigenvalues and eigenvectors of a similarity or affinity matrix to partition the data into clusters, making it effective for discovering non-linear and complex patterns in the data.

Algorithm for Spectral Clustering

Form	Form a distance matrix
Transform	Transform the distance matrix into an affinity matrix A.
Compute	Compute the degree matrix D and the Laplacian matrix L = D-A.
Find	Find the eigenvalues and eigenvectors of L.
Use	Use the 'k' largest eigenvalues from the previous step to project the matrix into a lower dimensional space of dimension 'k'.
Cluster	Cluster the data points.

Why use Spectral Clustering?

Flexibility in handling Complex data. This means that it can handle data of different shapes and structures.

Spectral Clustering is based on connectivity of data. This type of clustering is useful as it helps find meaningful relationships from the dataset.

Some cons of Spectral Clustering

Higher complexity of computation. As we need to compute eigenvectors and eigenvalues of Laplacian Matrix, the process might become computationally expensive.

Choosing optimal parameters. As we need to choose parameters like 'k', it might get hard to choose the most optimal value of 'k'.

Uses for Spectral Clustering

Image Segmentation: Can be used to group pixels or regions with similar characteristics. This is useful in important research fields like computer vision.

Social Network Analysis: Can be used to identify communities or groups of nodes with dense connections and understand relationships within these networks.

Recommendation systems: Spectral clustering can be employed in recommendation systems to group users or items based on their preferences or similarities

EIGENVECTOR CENTRALITY

What is it?

In graph theory, eigenvector centrality is the measure of the influence of a node in a network. If a graph node is connected to a node with a high eigenvector centrality score, the graph node in question gets a higher centrality score as well.

How to compute Eigenvector Centrality

Mathematically, eigenvector centrality is computed using the eigenvector corresponding to the largest eigenvalue of the network. The adjacency matrix represents the connections or relationships between nodes in a network. The eigenvector centrality is given by the corresponding entry in the eigenvector.

Considerations to make

Centrality score for nodes with no incoming relationships will converge to 0.

Due to missing degree normalization, higher degree nodes have a very strong influence on their neighbours' score.

Uses for Eigenvector Centrality

In academic research, eigenvector centrality is used in citation networks to identify influential papers or authors. Nodes with high eigenvector centrality represent papers that are frequently cited by other influential papers, indicating their impact within the research community.

Eigenvector centrality has been extensively applied to study economic outcomes, including cooperation in social networks. 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.

Connectivity of Graphs

Definition

Graph connectivity refers to the ability to travel or navigate between vertices in a graph

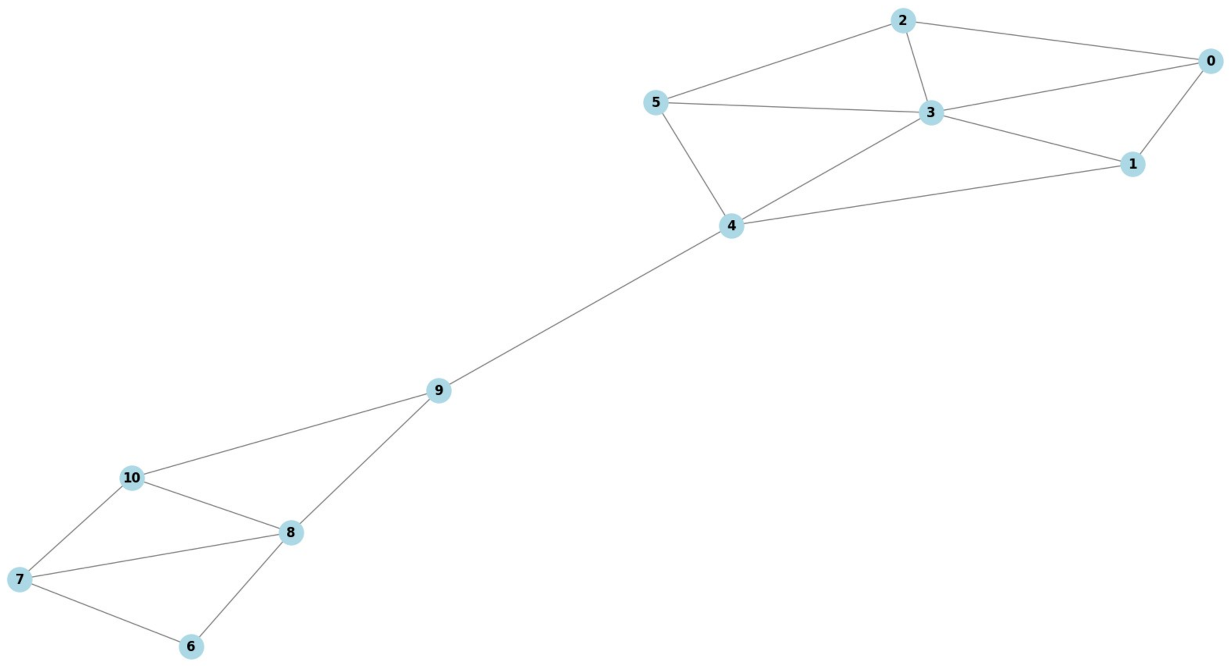
Classification

- ***Strong Connectivity***: A directed graph is strongly connected if there is a directed path between any two vertices in the graph.
- ***Weak Connectivity***: A directed graph is weakly connected if replacing all the directed edges with undirected edges produces a connected undirected graph.
- ***Disconnected***: No path is possible even by directed or undirected edges.

Graph connectivity with eigen values

- The connectivity of a graph can be determined using the Laplacian matrix. By calculating the eigenvalues of the Laplacian matrix and examining the second smallest eigenvalue, you can determine whether the graph is disconnected or has multiple connected components.
- Theorem: The second smallest eigenvalue $\lambda 2$ tells you about the connectivity of the graph. If the graph has two disconnected components, $\lambda 2 = 0$. And if $\lambda 2$ is small, this suggests the graph is nearly disconnected, with two components
- that are not very connected to each other.

Connected graph



Disconnected Graph

Applications of Laplacian Eigen Values

- **Graph Partitioning**: The eigenvalues of the Laplacian matrix can be used in graph partitioning algorithms.
- **Spectral Graph Theory:** The Laplacian matrix and its eigenvalues play a significant role in spectral graph theory. Spectral graph theory studies the properties of graphs using linear algebraic techniques.
- **Network Analysis**:The theorem can be applied in the analysis of various networks, such as social networks, transportation networks, and communication networks.

Overview of Laplacian Eigen Values

In the context of graph theory, the Laplacian eigenvalues are defined for a graph, which is a collection of vertices (nodes) connected by edges. The Laplacian matrix of a graph is a square matrix that encodes the graph’s connectivity information.

Types of Eigen Values for the Laplacian Matrix

- **Zero Eigenvalue:** As mentioned earlier, a Laplacian matrix always has a zero eigenvalue. It corresponds to a constant eigenvector, where all elements of the vector are the same.
- **Positive Eigenvalues:** The positive eigenvalues of the Laplacian matrix provide information about the connectivity and expansion properties of the graph. The magnitude of these eigenvalues indicates how fast information can spread or diffuse across the graph.

Types of Eigen Values for the Laplacian Matrix

- **Negative Eigenvalues:** In some cases, the Laplacian matrix may have negative eigenvalues. The presence of negative eigenvalues indicates that the graph has a non-trivial bipartite structure or a sign of imbalance in the graph’s connectivity.

Spectral Gap

- The difference between consecutive eigenvalues is referred to as the ”spectral gap.” The spectral gap provides insights into the connectivity and conductance properties of the graph. A larger spectral gap indicates a graph with stronger connectivity and better conductance between its components.

Uses

- **Graph partitioning and clustering**
- **Graph Drawing and Visualization**
- **Graph based image Segmentation**
- **Community Detection and Social Network Analysis**
- **Random Walks on Graphs**
- **Machine Learning and Data Analysis**
- **Network Resilience and Robustness**

THANK YOU

for more details, you can go through the pdf