Spectral Clustering Analysis

Network Statistics for Data Science (2AMS30)

Andrea Mangrella
Maarten van Sluijs
Roëlle Bänffer

Dataset Analysis:

For this project we were assigned the Network Science dataset:
It’s a graph in where ***edges*** are based on ***co-authorships*** in the area of Network Science, and the ***nodes*** **are the** ***authors***.

The dataset has a size of ***1461 nodes*** and a Volume of ***2742 edges***, making the average degree of the nodes ***3.7 edges each.***

Important properties of the Dataset:

- The graph is ***Unipartite*** and ***Undirected.***
- The edges are ***Unique*** and ***Unweighted.***
- There are ***no loops***.

The Connected Components Problem:

During last presentation we discussed an important property of the spectrum of a graph Laplacian matrix:

“The number of connected components is equal to the number of eigenvalues with real value 0.”

Problem: During the analysis of this dataset we immediately discovered that the graph contained ***more than 200 connected*** **components**, making solving the ***eigen problem*** with this many eigenvalues **computationally difficult**.

Solution: We then decided to **prune the graph** of all the smallest connected components (less than 20 nodes in size) and analyze each of the **remaining five**.

Eigen Gap:

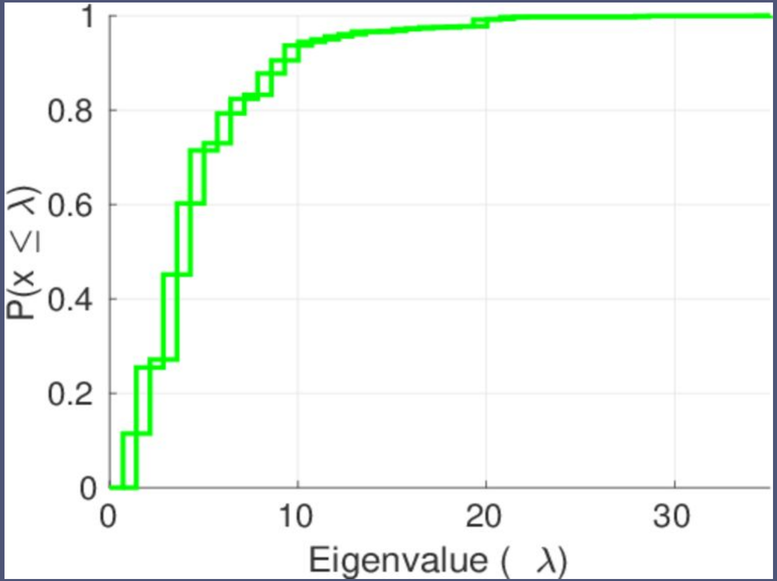
Another big problem we encountered during our analysis is how to decide the **optimal number of clusters** in our graph dataset.
We decided to use the Eigen Gap technique to find the number of clusters in the graph

The first step in the **Eigen Gap technique** consist in sorting the Eigenvalues in ascending order and then calculating the gap between consecutive eigenvalues.

The optimal number of clusters is often determined by identifying the first **significantly big gap.**

Spectral Plot of the Laplacian:

Spectral Plot with cumulative Eigenvalues distribution (*)



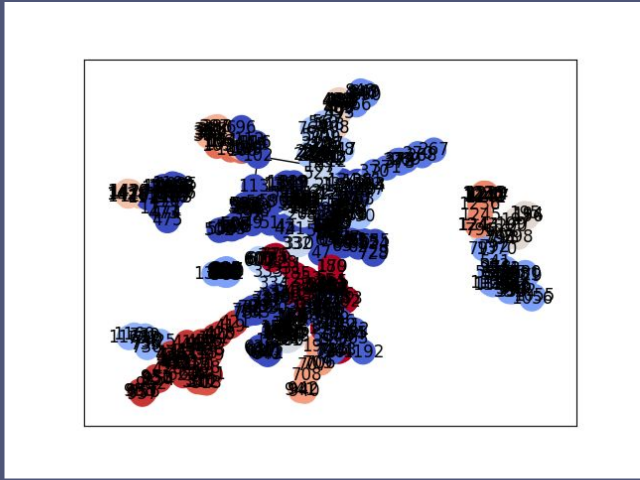
(*) Handbook of Network Analysis KONECT – the Koblenz Network Collection. Chapter 6, section 8

Spectral clustering methods

- Normalized spectral clustering with random walk
- Input: Similarity matrix S and number of cluster k
- Compute the normalized Laplacian Lrw (random walk)
- Compute the first k eigenvectors u1,...uk of Lrw
- …
- **Methods for clusterings**:
  - K-means clustering
  - Discretized
  - Cluster QR
- Output: Clusters

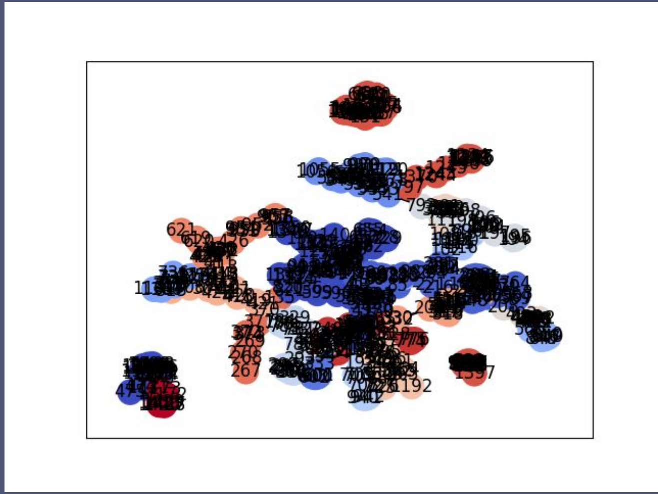
K-means clustering

- The center of each cluster is the mean of all the data points that belong to it
- Each data point belongs to the cluster with the nearest center point (mean)
- Minimizes squared errors on the ability to reconstruct neighbors
- K-means assume clusters are round within k-radius from centroid



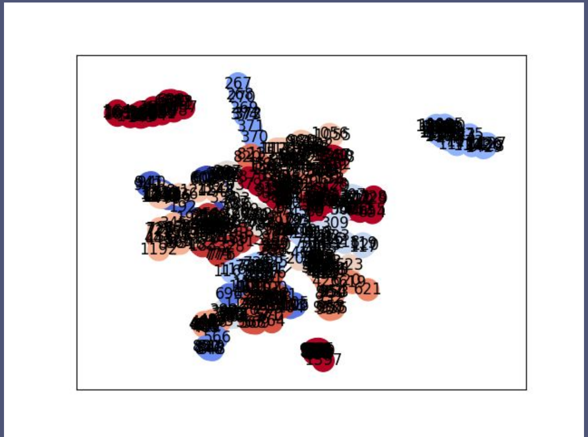
Discretized Clustering

- Minimize the uncertainty of the class variable conditioned on the discretized feature variable
- Continuous optima
- Makes use of orthonormal transformations of eigenvectors
- Goal:find transform that leads to discretization
- Less sensitive to random initialization



Cluster QR

- Extracts clusters from eigenvectors in spectral clustering
- Has no tuning parameters
- No iterations
- Can outperform k-means and discretization
- Is build for dealing with sparse SBM
- Use orthogonality to find cone center and point from different cluster



Evaluation Metrics

No labels/null model for the true community structure

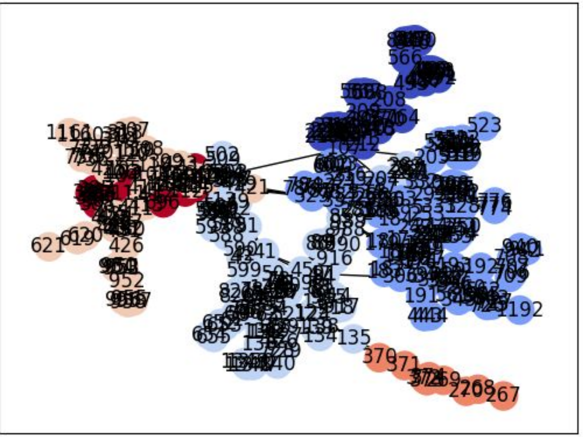
$$\text{Coverage} = \frac{(\text{\# of intra-community edges})}{(\text{total \# of comm edges})}$$

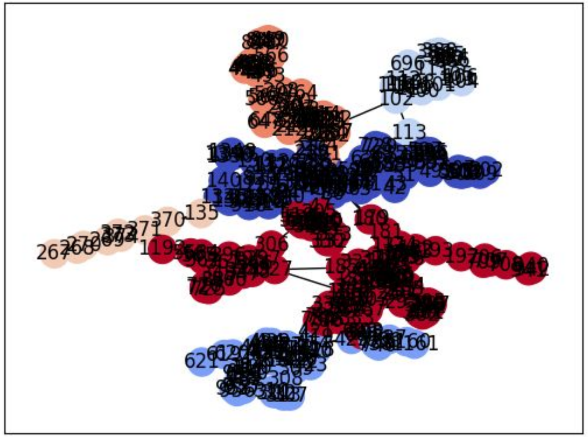
- Cluster edges are primarily to points in the same cluster

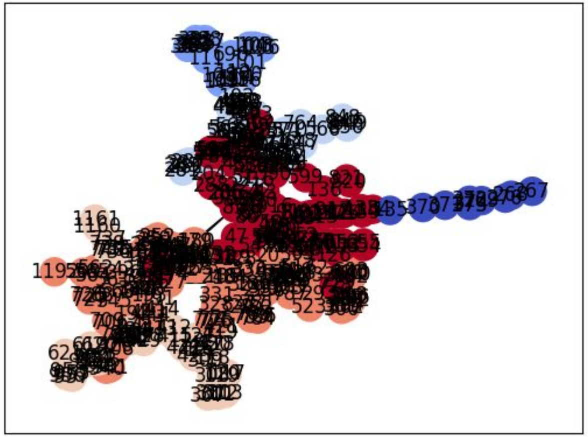
$$\text{Performance} = \frac{(\text{\# of intra-comm edges + \# of inter-comm non-edges})}{\text{Total \# of potential edges}}$$

- Points inside clusters are connected to each other, and not connected to points outside the cluster.

Results: Component 1 (N=379, k=6)

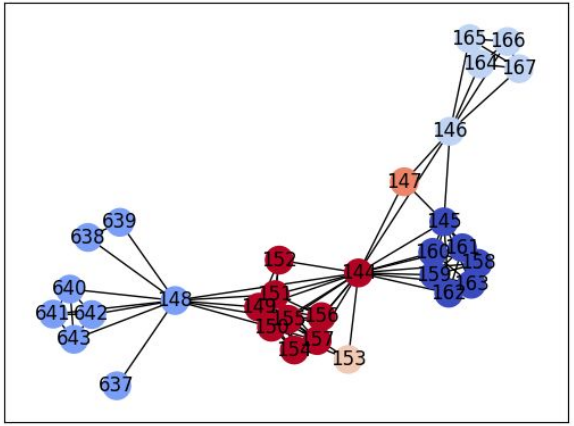


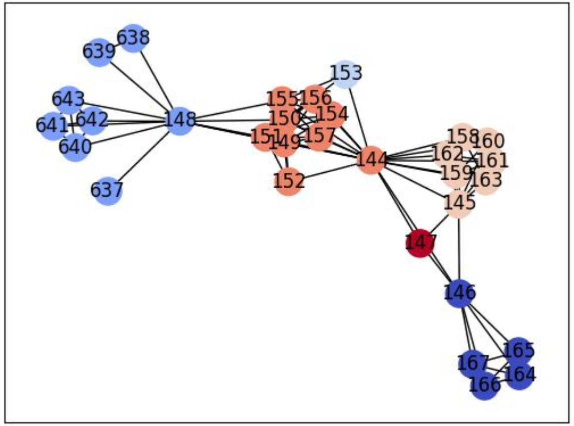


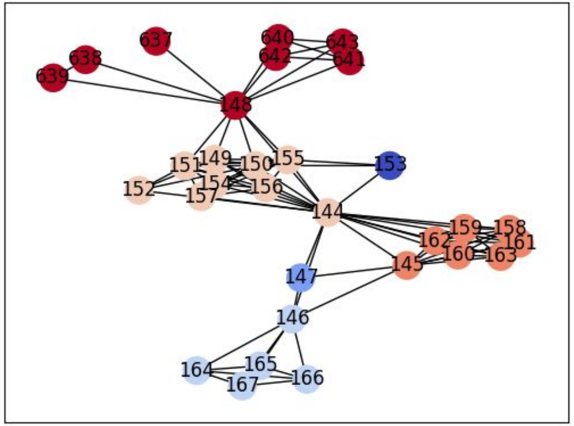


Method	K-means	Discretized	Clustering QR
(cov, per)	(0,98; 0,77)	(0,98; 0,78)	(0,98; 0,78)
dist	(120,118,59,49,24,9)	(120,110,59,55,25,10)	(120,110,59,55,25,10)

Results: Component 2 (N=31, k=6)

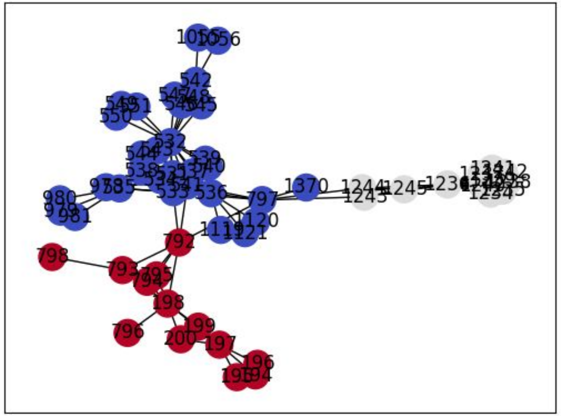


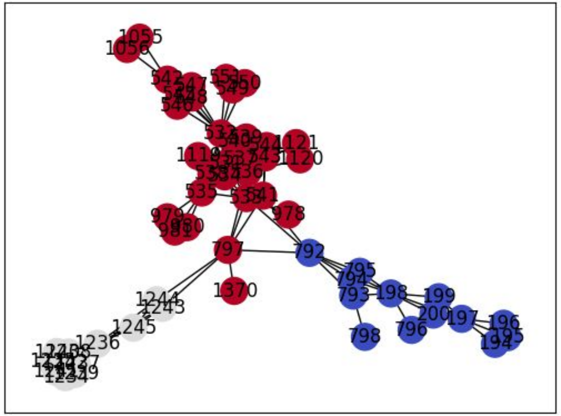


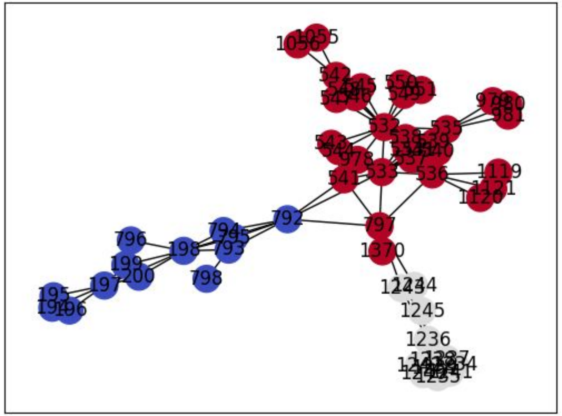


Method	K-means	Discretized	Clustering QR
(cov, per)	(0,84; 0,82)	(0,80; 0,83)	(0,75; 0,82)
dist	(16, 7, 5, 1, 1, 1)	(15, 7, 5, 2, 1, 1)	(15, 6, 5, 2, 2, 1)

Results: Component 3 (N=57, k=3)

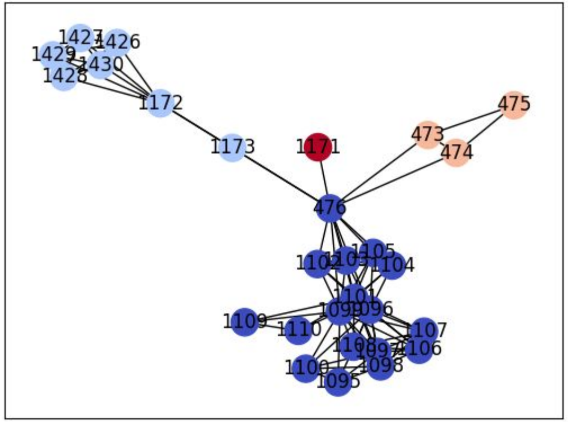


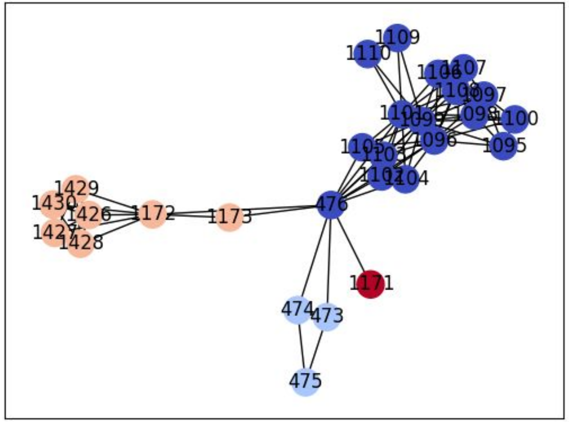


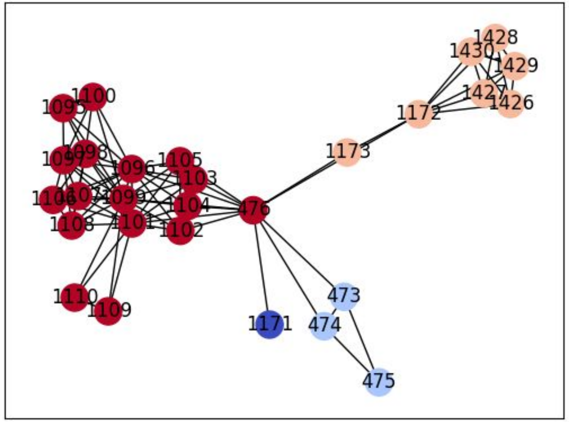


Method	K-means	Discretized	Clustering QR
(cov, per)	(0,97; 0,69)	(0,97; 0,69)	(0,97; 0,69)
dist	(32, 13, 12)	(32, 13, 12)	(32, 13, 12)

Results: Component 4 (N=28, k=4)

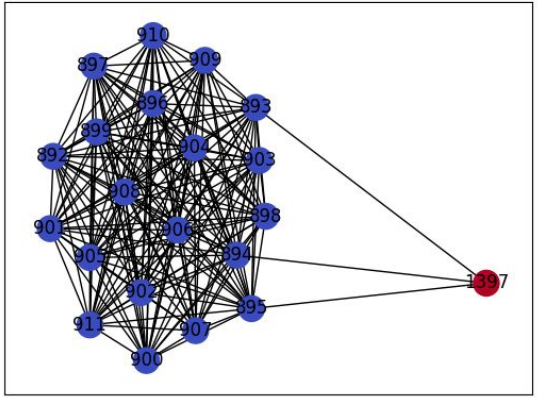


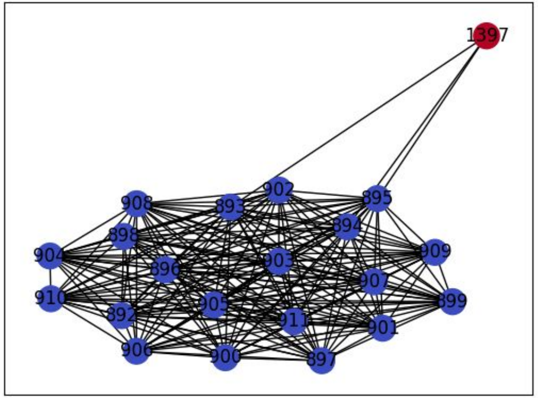


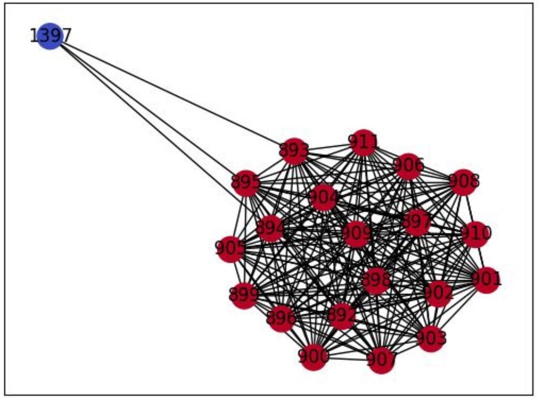


Method	K-means	Discretized	Clustering QR
(cov, per)	(0,95; 0,79)	(0,95; 0,79)	(0,95; 0,79)
dist	(17, 7, 3, 1)	(17, 7, 3, 1)	(17, 7, 3, 1)

Results: Component 5 (N=21, k=2)







Method	K-means	Discretized	Clustering QR
(cov, per)	(0,98; 0,99)	(0,98; 0,99)	(0,98; 0,99)
dist	(20, 1)	(20, 1)	(20, 1)

Main Takeaways

Can never truly know the underlying structure

However, metrics and visual inspection give some insight

Overall, method is quite effective
Some difficult/interesting cases:

- No easily spotted segmentation
  - Clustering QR does best here
- Single “disconnected points”
  - Likely due to the difficulty of selecting the right k