# Homework 4
# Graph Spectra

Badai Kesuma Hamid Dimyati

November 2020

## 1. Code commentary

The implementation of graph spectra is done in Python mainly using `KMeans` from `sklearn` library, `networkx`, `numpy`, `scipy`, and `matplotlib` library. The application clusters the given graphs using the algorithm as described in the paper "On Spectral Clustering: Analysis and an algorithm" [1]. This application is capable of finding the optimum number of *k* where the eigenvalues start dropping and clusters the data into *k* clusters via K-means.

Within the jupyter notebook (`experiment.ipynb`), we start by reading the data files which represent a list of edges. Then, we generate a graph object using `networkx` library and plot the graph to see a big picture of the network. Next, we construct an Affinity matrix from the data, and generate a diagonal matrix D. After that, we continue by build a normalized Laplacian matrix L ($L = D^{-½} A D^{-½}$), to generate eigenvectors and respective eigenvalues using `scipy` library.

Before we perform a clustering task, we first need to define the optimum *k* by investigating which one is the largest eigengap (the difference between two consecutive eigenvalues). Once we obtain the optimum *k*, next we extract the eigenvectors associated to the *k* largest eigenvalues. Then, we do normalization towards that matrix and the result is being used to run the K-means clustering.

We could also see how many optimum clusters could be obtained from the data by checking the ratio cut. This kind metric could be achieved by looking at the second eigenvector (Fiedler vector) which corresponds to second smallest eigenvalue from the Laplacian matrix L ($L = D - A$).

The algorithm is implemented on two data sets: a medical innovation data set [2] in data/example1.dat and a synthetic graph in data/example2.dat.

## 2. How to run

Make sure all the required libraries are installed. Run the Jupyter Notebook on the terminal to open the browser. Open the `experiment.ipynb` from the zip file. Reader should simply follow the Jupiter Notebook cell per cell.

## 3. Results

In the beginning, we load the graph from example1 and example2 into tuples of values that represent edges. Using the above algorithm, we can find the optimum $k$ with the largest eigengap in the sorted eigenvalues, with $k = 4$ in example1 and $k = 2$ in example2. We, then, can see the result in Figure 1, the clusters using K-means clustering with the number of clusters = $k$.

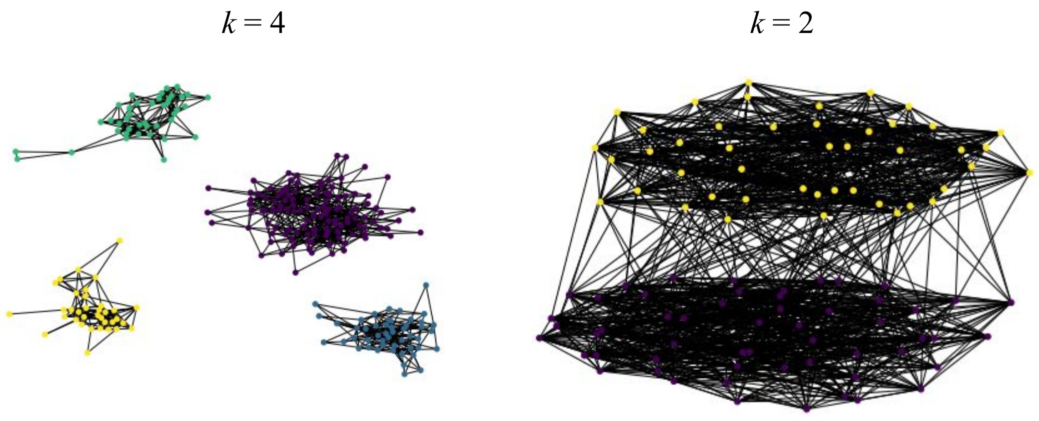

Figure 1. Graphic clustering results of example1 (left) and example2 (right)

In order to see the communities in the graph, we have to plot the Fiedler vector, the eigenvector corresponding to the second smallest eigenvalue of the Laplacian matrix. From the sparsity pattern in Figure 2, we can clearly see the 4 communities in example1 with 1 community that has bigger size compared to the others. But it is hard to see the communities in example2 because the nodes are connected to each other in between the 2 communities. For Fiedler vector results could be seen in Figure 3. The line represents the communities in the graphs. It could be concluded that graph spectra compute a reasonable clustering.

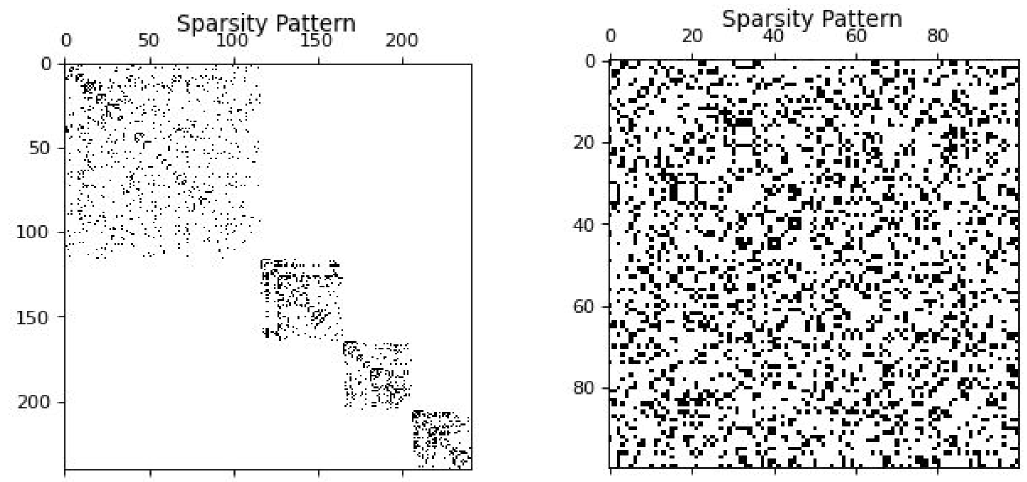

Figure 2. Sparsity Affinity matrix of example1 (left) and example2 (right)

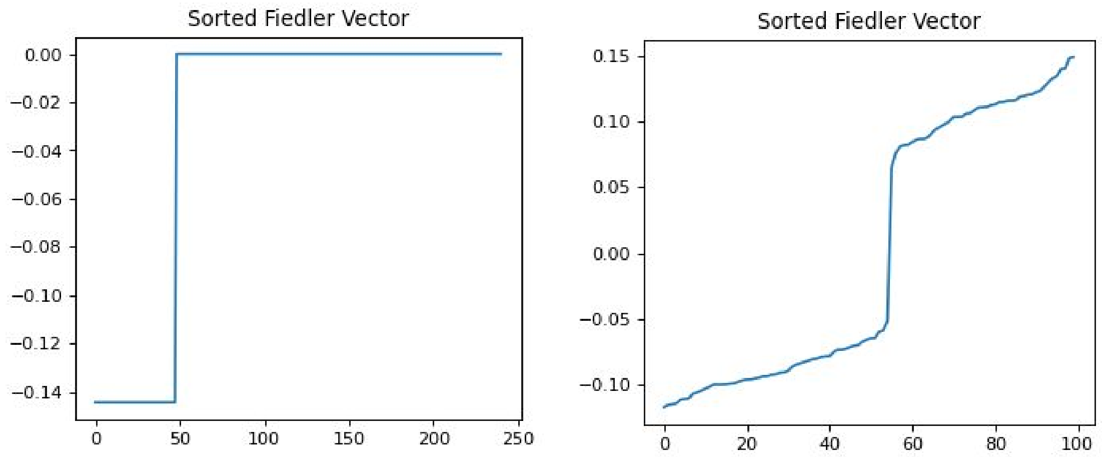

Figure 3. Fiedler vector of example1 (left) and example2 (right)

## References

[1] Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. In Advances in neural information processing systems (pp. 849-856).

[2] Coleman, J. S., Katz, E., & Menzel, H. (1966). Medical innovation: A diffusion study. Indianapolis: Bobbs-Merrill Company.