Data Mining, ID2222

Short report on lab assignment 4

Homework 4: Graph Spectra

Ivan Klabucar and Simon Langrieger

December 6, 2021

1 Analysis of example network

In this section we will analyze a custom network with approaches described during our lectures. This network was designed so that it would have two strongly connected components, but in such a way that the larger component is split into two distinct clusters. The network is shown on Figure 1.

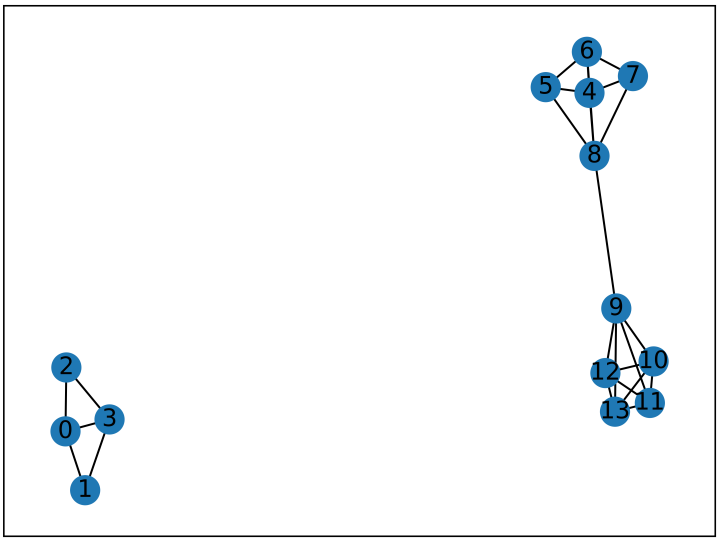

Figure 1: my_example.dat

1.1 Eigenvectors of similarity matrix

First we computed the eigenvectors of a normalized adjacency matrix A.

$$A_{ij} = 1/\deg(node_i), \text{ for every edge ij else 0}$$

The 3 largest eigenvalues were as follows:

$$\lambda_1 = 1.0000 \tag{1}$$
$$\lambda_2 = 1.0000 \tag{2}$$
$$\lambda_3 = 0.9231 \tag{3}$$

And their corresponding eigenvectors:

$$v_1 = \begin{bmatrix} -0.5000 \\ -0.5000 \\ -0.5000 \\ -0.5000 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad v_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.3162 \\ 0.3162 \\ 0.3162 \\ 0.3162 \\ 0.3162 \\ 0.3162 \\ 0.3162 \\ 0.3162 \\ 0.3162 \\ 0.3162 \end{bmatrix} \quad v_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.3515 \\ 0.3458 \\ 0.3515 \\ 0.3458 \\ 0.2547 \\ -0.2192 \\ -0.3165 \\ -0.3165 \\ -0.3165 \\ -0.3165 \end{bmatrix} \tag{4}$$

The two largest eigenvalues are both 1. Eigen gaps of 0 indicate disconnected components, and in our case this is supported by our graph. The third eigenvalue is especially interesting because even though it is not 1, it is very close to one. This can be interpreted as the existence of a cluster that's very poorly connected to the rest of the graph. This is of course the cluster composed of nodes 9, 10, 11, 12, 13, and 14 which are connected to the rest of the graph with only a single bridge between 9 and 8. Another thing to note is that in the third eigenvector the distinct clusters can be identified by their corresponding values in the eigenvector. The cluster around node 6 got assigned positive values while the cluster around 10 got assigned negative values.

1.2 Eigenvectors of the Laplacian

If we repeat the same experiment with the Laplacian matrix we get similar results. This time however, we inspect eigenvectors with the smallest values.

$$\lambda_1 = 0.0000 \tag{5}$$
$$\lambda_2 = 0.0000 \tag{6}$$
$$\lambda_3 = 0.2984 \tag{7}$$

And their corresponding eigenvectors:

$$v_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -0.3162 \\ -0.3162 \\ -0.3162 \\ -0.3162 \\ -0.3162 \\ -0.3162 \\ -0.3162 \\ -0.3162 \\ -0.3162 \\ -0.3162 \end{bmatrix} \quad v_2 = \begin{bmatrix} 0.5000 \\ 0.5000 \\ 0.5000 \\ 0.5000 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad v_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.3336 \\ 0.3336 \\ 0.3336 \\ 0.3336 \\ 0.2341 \\ -0.2341 \\ -0.3336 \\ -0.3336 \\ -0.3336 \\ -0.3336 \end{bmatrix} \tag{8}$$

Just like in the last section, the second eigenvalue is also zero just like the first one. This also indicates a disconnected component. The third eigenvalue is interesting because it is close to zero. This implies its eigenvector describes a cluster that is very poorly connected to the rest of the graph. Virtually the same inference can be made as in the last subsection.

1.3 Eigenvalues on real-world graphs

In preparation for the main task of the assignment, which is finding clusters in example real-world graphs, we calculate their adjacency matrix' biggest eigenvalues. This hopefully gives us an intuition about the value for the number of clusters we are looking for.

We looked at two graphs which are simply called "example1" and "example2". In example1 we find that the four biggest eigenvalues are 1.0 which tells us that the graph consists of 4 disconnected clusters. Therefore we will choose $k = 4$ for example1.

For example2 we have only one eigenvalue being 1.0, hence there are no disconnected communities in the graph. However, we see that after the second eigenvalue ($\lambda_2 \approx 0.83$) the magnitude of the third eigenvalue clearly drops significantly to around 0.33. This tells us that there are probably two clusters in the graph which are loosely connected, which is why we go for $k = 2$ for example2.

2 Graph clustering algorithm

In Figure 2 and Figure 3 we plotted example1 and example2 respectively using the python libraries networkx and matplotlib. The plots clearly support our proposed values for the number of clusters found in the last section. This is however only really possible in smaller examples which is why used the method with the eigenvalues nevertheless.

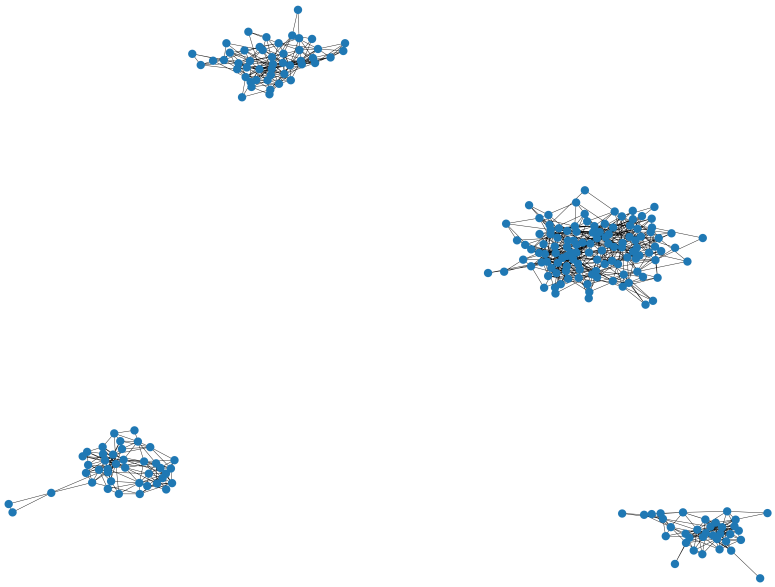

Figure 2: example1.dat

2.1 Spectral clustering

Now to the actual algorithm which finds the clusters in graphs. The algorithm was originally designed to robustly cluster n points in an d-dimensional space into k clusters. The input to the algorithm is a nxn dimensional matrix A with $a_{ij} = \exp(-d(s_i, d_j))$ with $d(s_i, s_j)$ being an distance measure between points s_i and s_j. Hence if two points are infinitely far away from each other a_{ij} will be 0.

Now we work on graphs which we represent using adjacency matrices. However we can just view the adjacency matrix as the aforementioned distance matrix A where disconnected nodes are infinitely far apart and all connected nodes have the same distance to another, which results in the ones in the distance matrix.

Then we define D as a diagonal matrix with the sum of each row of A on the

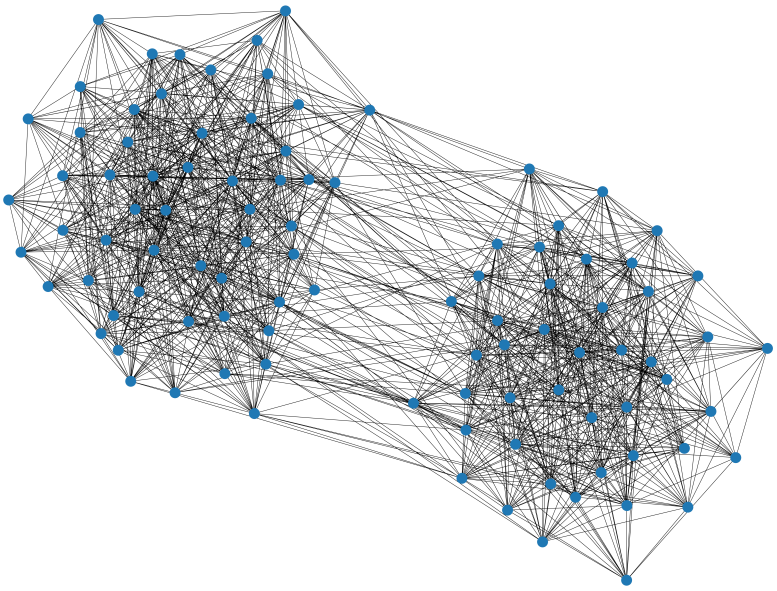

Figure 3: example2.dat

diagonal to finally get a matrix L by $L = D^{-0.5}AD^{-0.5}$.

After that, we calculate the k biggest eigenvectors of the matrix L, with k being the number of clusters we want to find. We construct a matrix X by stacking these eigenvectors in columns. Which gives us an nxk dimensional matrix. Every row of X is then re-scaled to have a unit length of one. This normalised matrix is called Y.

Every row in Y is then interpreted as a point in an k-dimensional space. Finally we run any implementation of a k-means algorithm on these points y and assign node j to the cluster c only if the point y_j is assigned to the cluster c. This is then our final clustering.

2.2 Results

We finally ran this algorithm on our example graphs and the results can be seen in Figure 4 and Figure 5. We see that we were successfully able to cluster the graphs into the expected communities.

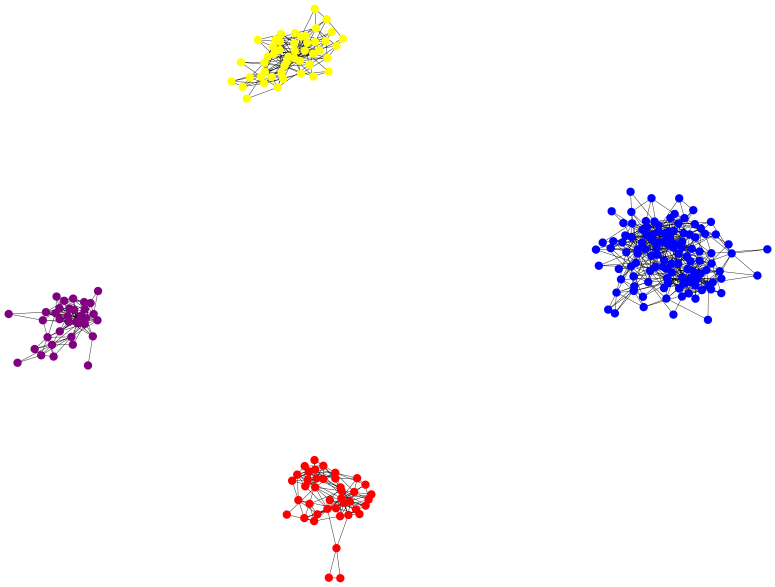

Figure 4: example1.dat Sorted into clusters with clustering algorithm

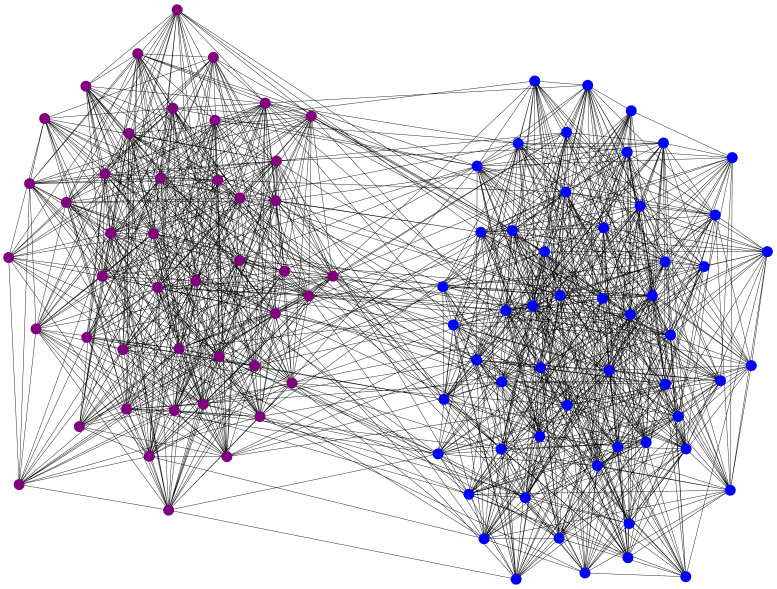

Figure 5: example1.dat Sorted into clusters with clustering algorithm