BINARY CLUSTERING OF GRAPH VERTICES

ABSTRACT

PURPOSE: To explore the use of Fiedler vector to produce clusters for a given edge list dataset.

METHODS: A Fiedler vector clustering algorithm is explored in this report. It uses the edge lists of simple undirected graphs to find their Adjacency and Laplacian Matrices, and using those, find the Fiedler vector to group vertices of the graph in two clusters.

RESULTS: The results clearly demonstrate the effectiveness of the algorithm as shown by the minimal no. of edges between the clusters, and a very even clustering

CONCLUSIONS: This algorithm is effective at producing a binary clustering of huge real-world datasets with minimal outliers and appropriate dimensions. It was particularly effective at producing proper clusters for the given dataset.

Word Count: 118

INTRODUCTION

The purpose of this exploration was to use the Fiedler method to produce data clusters from a given edge list dataset and interpret the results using previous knowledge.

The Fiedler vector is the eigenvector of the second smallest of eigenvalue, which is found by computing the Laplacian matrix of the graph from its adjacency matrix, and then finding its second smallest eigenvector. The binary clustering using the Fiedler vector is then done by assigning positive and negative one values to the vertices and clustering them as such.

The question being explored is to cluster the vertices of a graph, given its edge list dataset, by using the Fiedler vector method. We do not have a testable hypothesis as we aren't conducting a meta-analysis of the results which would give a measure of accuracy to the algorithm.

METHODS

The clustering algorithm being explored in this report is one based on using the edge lists of simple undirected graphs to find their Adjacency and Laplacian Matrices, and using those, finding the Fiedler vector to group vertices of the graph in two clusters.

Three functions were used to create the output of a1_20292366.m. Functions plot271a1(Amat, cvec) and xy = circlen(n) were supplied by the instructors of CISC 271. They were used to graphically emulate the clusters using the input sets A and set12 supplied by the function in focus, function set 12 = a1_20292366(elist).

Function set 12 = a1_20292366(elist) takes in an edge list and outputs a vertex clustering in terms of 2 sets where sets set1 and set2 contains indices of vertices with value -1 and value +1 respectively. The function initializes the size of adjacency matrix A with zeroes using the vertex with the highest numerical value. It then uses a 'for' loop to go replace the zeroes in all cell intersections of vertices, that have edge connections, with ones.

The function then diagonalizes a column vector which is the sum of each row in the adjacency matrix to create a degree matrix. After calculating the Laplacian matrix by subtracting the adjacency matrix from the degree matrix, it finds the eigenvectors of the Laplacian.

The final calculation part of the function follows it extracting the Fiedler vector from the eigenvector matrix, which contains values equal to zero or one. To make it more suitable for clustering, since zero is strictly neither positive nor negative, we make replace all the zero values with negative ones.

The 'set12' or the manipulated version of the Fiedler vector, is then separated into two sets to make the display output clearer. This is done by initializing two row vectors of size 10, looping through the Fiedler set12, and putting all the indices with negative one into set1 and positive one into set2 respectively. This may be inaccurate for datasets with uneven clustering, but since the algorithm assumes even clustering, a vector of size 10 was used instead of a bigger vector which might have taken up more space.

Finally, the redundant zeroes in the set are removed and the output sets displayed in the command window, along with the cluster graphs in a separate figure window.

The code was tested throughout the function by having the initial matrices and vectors be displayed. It was also tested using the given 'testedges' and 'testset' text files which had their output outlined in the given Assignment pdf. No statistical tests were conducted to test the accuracy of this output, instead the accuracy of the results is discussed in the discussions section.

RESULTS

Table 1 contains the two-output vertex sets. Figure 1 outlined below shows the graph clustering for the edge list provided in file 20as304.txt.

Table 1: Clusters of vertices for the 20as304.txt graph where each row of the text file contains connections between the vertices of the cluster

SET	**VERTICES**									
1	1	2	3	5	7	12	16	17	19	20
2	4	6	8	9	10	11	13	14	15	18

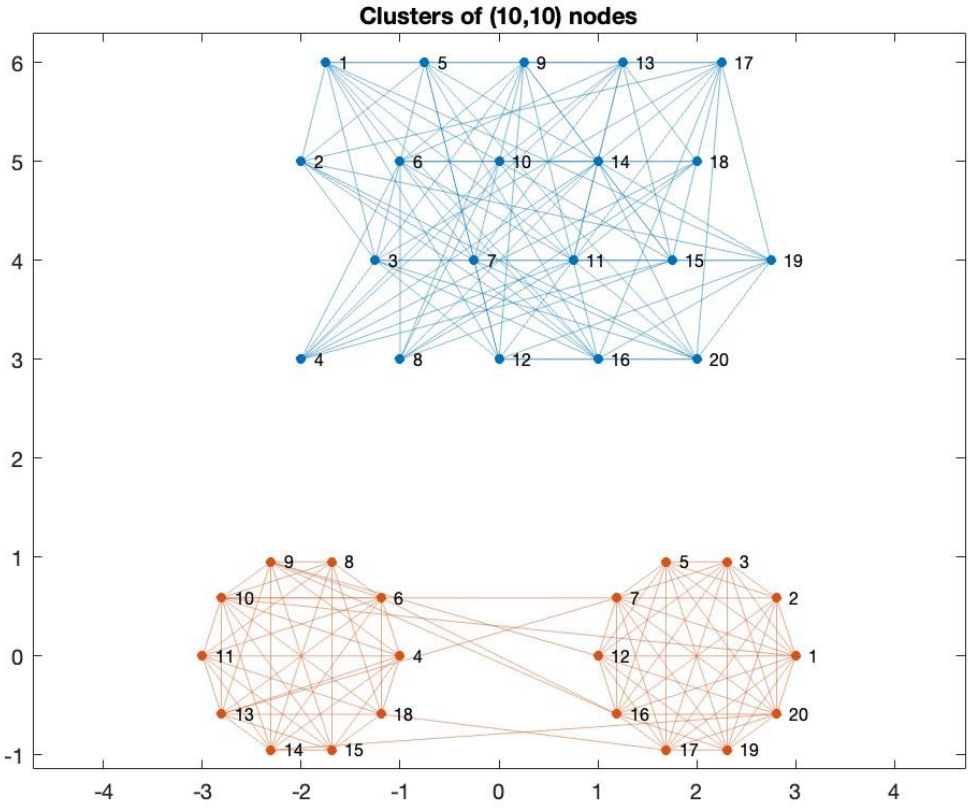

Figure 1: Diagrams of the 20as304.txt graph. The upper section of the graph shows the mapping of the graph in a rectilinear grid. The lower section of the graph shows the clustering of the vertices as computed by the Fiedler vector

DISCUSSION

The results section clearly shows two distinct clusters with 8 edges between them. We can make a few observations from the output in the results section shown above, the first of which can be gleaned with a cursory glance.

A small no. of edges between clusters is a typical attribute of a good clustering, which is relatively true for our output as there are only 8 edges between the vertices in sets set1 and set2, relative to the 20 vertices which have close to 180 edge connections between them. So, approximately 4% of all connections exist between the clusters, which qualifies this to be a good clustering.

Another attribute of good clustering is that the size of the clusters turns out to be similar, which seems to be the case here, as both sets, set1 and set2, have 10 vertices each. This, in conjunction with the fact that there is a good separation between the clusters speaks to the effectiveness of this algorithm for such data sets.

Even though the algorithm gives us a good clustering for the given dataset 20as304.txt, it is not guaranteed that it would work for all datasets. For example, a disconnected graph would likely produce multiple clusters that may be less meaningful as this algorithm relies on the connectivity of simple graphs.

It also would not be effective at producing clusters for datasets with a lot of outliers present as it would construct an inaccurate no. of clusters which may be unusable for real world interpretation. It would also only be effective for binary clustering which means that it would not be a great algorithm for datasets that contain more than 2 dimensions.

Though the reasons above argue that this method is a mostly ineffective method of clustering real-world data, this algorithm can process a huge amount of data from large spanning graphs.

We can conclude that this algorithm is effective at producing a binary clustering of huge real-world datasets with minimal outliers and appropriate dimensions, not limited to the one given in the dataset – 20as304.txt.