Problem Set 5

This homework you will use some of the dimensionality reduction techniques we have learned in class and apply it on a well-known problem which is graph drawing. You will not be responsible for the graph drawing part as that will be provided for you in the attached script. You will be responsible for the parts that are related to the material we covered in class.

Your code must be all written within the file HW5.py provided to you. The drawing part is also provided in the file.

Part 1 (5 points) MDS-based node position

Implement the following algorithm:

Algorithm for position of the graph nodes

Data: A weighted graph G(V,E). If the graph is unweighted, assign a weight of 1 to every edge.

Result: a dictionary that assigns to every node v in V a position np.array(x,y).

1-Compute the distance matrix D of the graph G

2-Use sklearn MDS function on the distance D and specify the *n_components*=2.

Make sure your input is a distance matrix and not a point cloud (the default is a pointcloud). See this for more details. In other words, you need to select `dissimilarity='precomputed'` *as part of your input to the mds function.*

3- Get the position you obtained from MDS and save them in a dictionary as illustrated in the attached file (HW5.py)

4- Return the dictionary of the positions

After you finish the algorithm, run the function "draw_graph" provided for you on the example graph provided for you and the output you obtained from the above algorithm and test your results. Save the image you obtained.

Part 2 (5 points) Laplacian-based node position

Implement the following algorithm:

Algorithm for position of the graph nodes

Data: A weighted graph G(V,E). If the graph is unweighted, assign a weight of 1 to every edge.

1-Compute the distance matrix D of the graph G

2-Use sklearn spectral embedding function on the distance D and specify the *n_components* =2.

Again here you need to make sure you input a dissimilarity matrix.

3- Get the positions you obtained from spectral embedding output and save them in a dictionary as illustrated in the attached file (HW5.py)

4- Return the dictionary of the positions

After you finish the algorithm, run the function "draw_graph" provided for you on the example graph provided for you and the output you obtained from the above algorithm and test your results. Save the image you obtained.