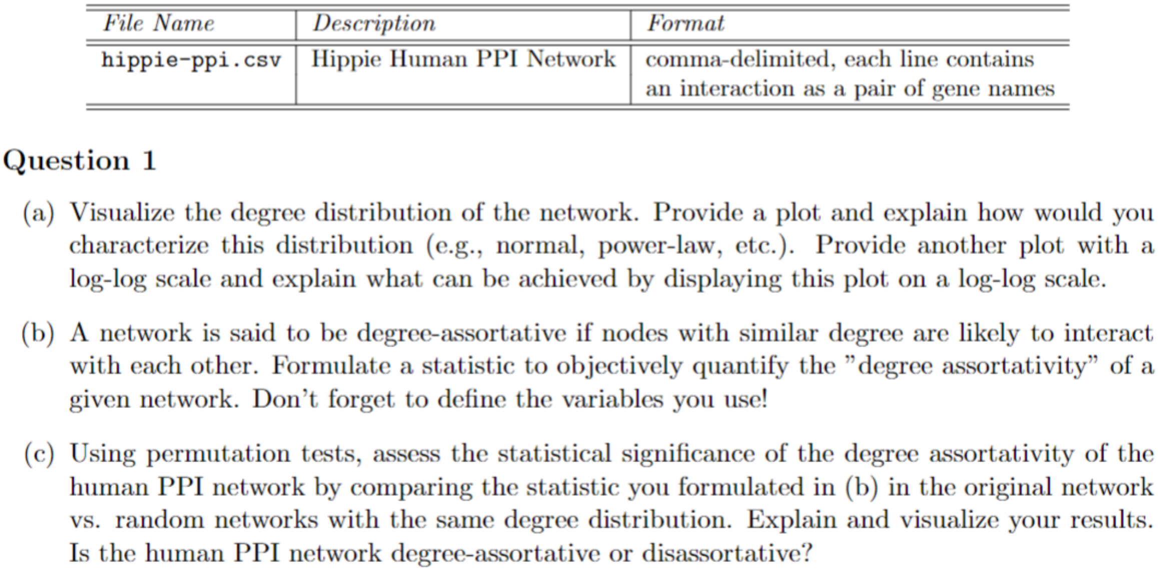
Harshita Kumar (hxk613) and Mehlam Saifudeen (mms330)



File Name	*Description*	*Format*
`hippie-ppi.csv`	Hippie Human PPI Network	comma-delimited, each line contains an interaction as a pair of gene names

Question 1

(a) Visualize the degree distribution of the network. Provide a plot and explain how would you characterize this distribution (e.g., normal, power-law, etc.). Provide another plot with a log-log scale and explain what can be achieved by displaying this plot on a log-log scale.

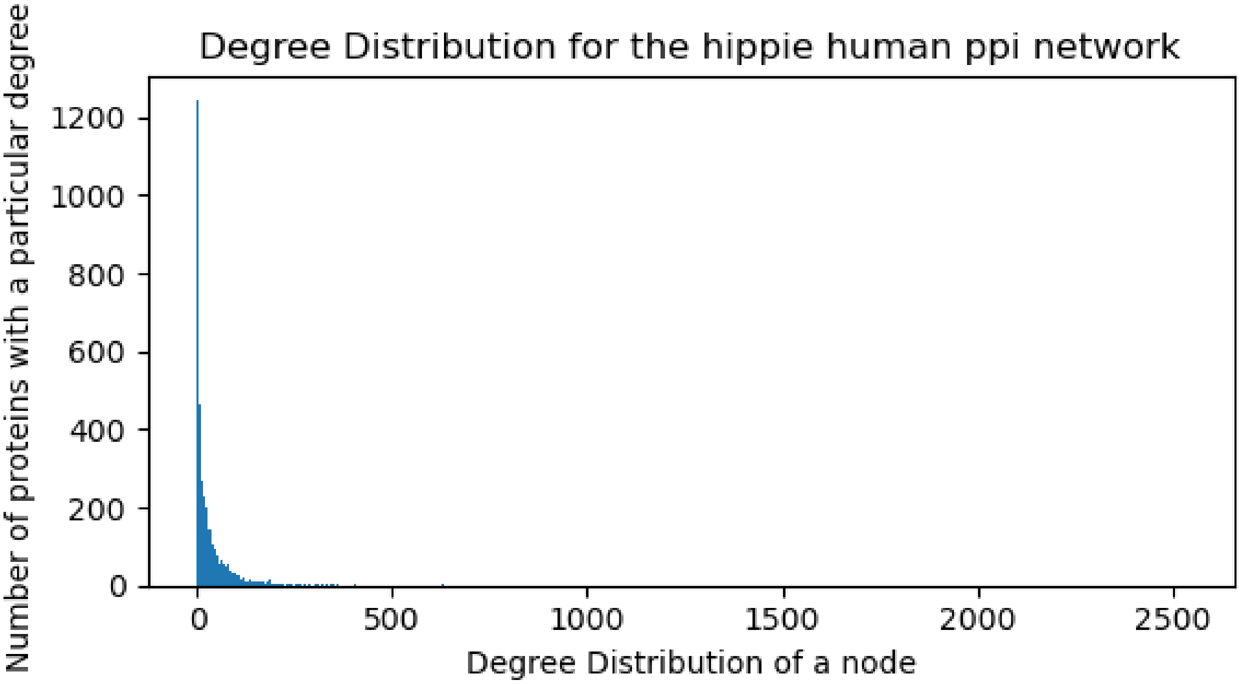
(b) A network is said to be degree-assortative if nodes with similar degree are likely to interact with each other. Formulate a statistic to objectively quantify the "degree assortativity" of a given network. Don't forget to define the variables you use!

(c) Using permutation tests, assess the statistical significance of the degree assortativity of the human PPI network by comparing the statistic you formulated in (b) in the original network vs. random networks with the same degree distribution. Explain and visualize your results. Is the human PPI network degree-assortative or disassortative?

Ans.

a. The code snippet helps obtain the plot below which shows the degree distribution of the hippie human ppi network.

```
#Question 1a part 1
plt.hist(degree_sequence,bins=range(min(degree_sequence),max(degree_sequence)+1))
plt.xlabel('Degree Distribution for the network')
plt.ylabel('Number of interactions for each protein')
plt.title('Degree Distribution for the hippie human ppi network')
plt.gca().set_aspect('equal', adjustable='box')
plt.show()
```

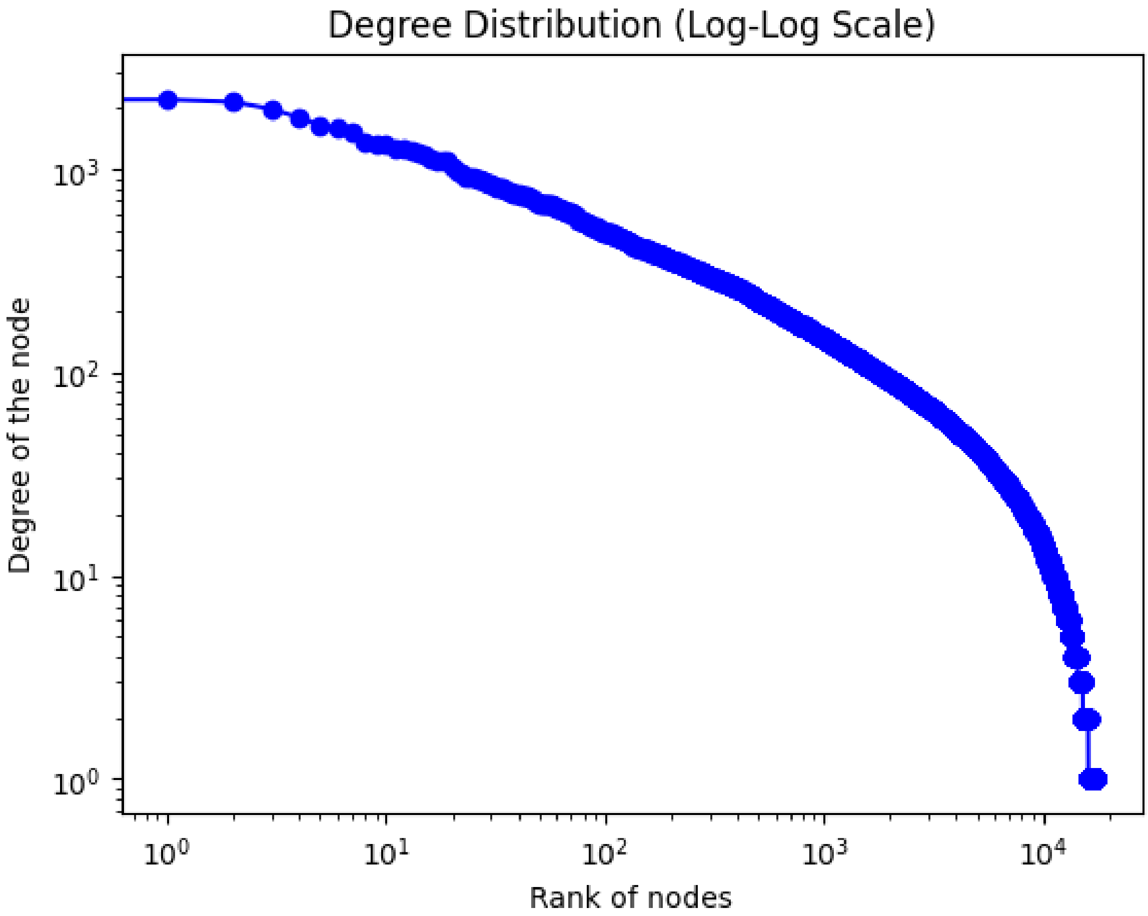


We can characterize this distribution as a power law distribution. This is because the mathematical relationship seen in the graph above, between the two variables which are the number of interactions and the degree distribution, follows the form of a power function. If a distribution of the network consists of a few nodes with a high degree (many connections) and many nodes with low degree (few connections). Mathematically, the power law can be described as: P(n) = c * n^(-a), where P(n) is the probability that a node in the network has a degree n, a is the power law exponent, and c is the normalizing constant. If the value of a is less than 3, it can be concluded that it follows the power law distribution.

From the graph above we see that there are 1244 proteins with only one degree of interaction with other proteins, whereas there is only one protein with 2530 degrees of interactions.

Below is the code snippet that generates the same plot but on a log-log scale. Here the x-axis represents the rank of nodes in the network which have been sorted in descending order by degree, and the y-axis represents the degree of the node.

```
    #Question 1a part 2
    plt.loglog(sorted(degree_sequence,    reverse=True),    'b-',
marker='o')
    plt.xlabel('Rank')
    plt.ylabel('Degree')
    plt.title('Degree Distribution (Log-Log Scale)')
    plt.show()
```



Plotting the degree distribution of a network in a log-log scale can help us identify the presence of a distribution characteristic such as power-law or exponential distribution etc depending on slope of the line seen in the graph. From the plot above we can see that it suggests an exponential distribution. This is because as the rank of the nodes increases in the list, the degree of the node decreases which indicates that this graph follows an exponential decrease. The presence of an exponentially decreasing curve on the log-log scale of a protein protein interaction network typically indicates a power law distribution. This can be seen in the mathematical equation of the power law. It suggests that a small number of highly connected "hub" proteins interact with many other proteins, while most proteins have relatively few connections. Such scale free networks are observed a lot in various biological systems, including ppi networks. The presence of central proteins in the network can be used to indicate robustness and stability to the system. At the same time, it also indicates efficient communication between different components of the biological system.

b. A popular statistic to objectively quantify the degree assortativity of a network is the Pearson correlation coefficient. The Pearson correlation coefficient measures the linear association between two variables.
In the case of the degree assortativity, the two variables are the degrees of the nodes that are connected by a single edge. The Pearson correlation coefficient of the degrees of nodes connected by edges in the network provides a measure of the degree assortativity of the network. A positive value of the Pearson correlation coefficient indicates that nodes with similar degree are more likely to interact with each other, whereas a negative value would indicate that nodes with dissimilar degree are more likely to interact with each other.
The variables for the statistic to quantify the degree assortativity of the network:
d = degree of the node
e = edge connecting two nodes in the network
(d_i, d_j) = the degrees of the nodes connected by edge e
n = the number of edges in the network
Σ(d_i - mean_d)^2 and Σ(d_j - mean_d)^2: the sum of the squares of the deviations of the node degrees d_i and d_j from the mean degree mean_d of all nodes in the network.

The statistic can be formulated as **r = [Σ(d_i - mean_d) * Σ(d_j - mean_d) / Σ(d_i - mean_d)^2 * Σ(d_j - mean_d)^2]**

Here **r** is the Pearson correlation coefficient. A value of r that is close to 1 indicates a high degree of degree assortativity whereas a value of r that is closer to -1 indicates a high degree of disassortativity and a value that is 0 indicates no degree assortativity.

The code snippet below gives the Pearson correlation coefficient value:

```
#Question 1b
node_degrees = np.array([(G.degree(e[0]), G.degree(e[1]))
for e in G.edges()])
degree_assortativity = np.corrcoef(node_degrees[:, 0],
node_degrees[:, 1])[0, 1]
```

```
print(degree_assortativity)
```

The value of r obtained for the human protein protein interaction network using the Pearson correlation coefficient as a statistic to quantify the degree assortativity is `-0.05050037662470296.` This value is closer to -1 hence indicating that high degree of disassortativity. This means that the protein protein interaction network consists of high degree nodes connected to some low degree nodes and at the same time since the value is a very small negative number it can also be said that there are a number of random connection patterns in the remainder of the network.

c. Permutation tests can be used to assess the statistical significance of the degree assortativity of a network by comparing the Pearson correlation coefficient formulated for the network as a statistic above against the coefficient of some random networks with the same degree distribution.
The following are the steps that were done for the permutation tests:
1. The Pearson correlation coefficient (r) for the human ppi network was obtained from earlier in part b.
2. Generate 5 random networks (can do many more but we were not able to due to long run time of the code) with the same degree distribution as the human ppi network. A function was defined called generate_random_network that leveraged the built-in built function called configuration_model of networkx which is used to generate random graphs with a given degree which in this case would be the degree sequence of the nodes in the protein protein interaction network given to us.
3. For each random network generated using the same degree as the protein protein interaction network, the Pearson correlation coefficient was calculated.
4. Calculate the p value as the fraction of random networks with a Pearson correlation coefficient greater than or equal to the coefficient of the human ppi network
5. If the p value is greater than 0.05 then we can conclude that the network is significantly degree assortative or disassortative.

The code snippet below shows the permutation test in python:

```
# Generate a large number (e.g., 5) of random networks
random.seed()
num_random_networks = 5
random_network_r_values = []
```

```
    for i in range(num_random_networks):
        G_rand = generate_random_network(degree_sequence)
        r_rand = nx.degree_assortativity_coefficient(G_rand)
        random_network_r_values.append(r_rand)

    # Convert the list of random network Pearson correlation
    coefficients to a pandas DataFrame
    df2 = pd.DataFrame({'r_values': random_network_r_values})
```

Below are the Pearson correlation coefficient values of the 5 random networks generated from the same degree distribution as the human ppi network.

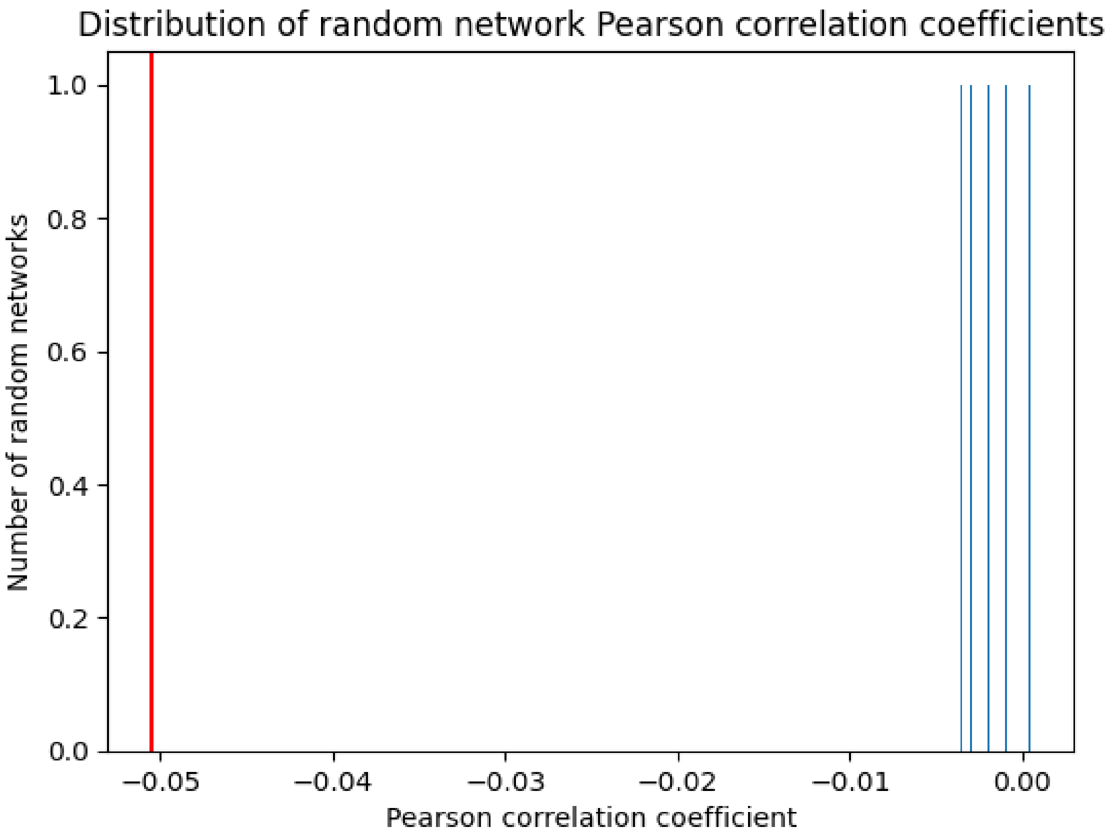
```
r_values
0 -0.000259
1  0.000459
2 -0.000563
3 -0.001875
4 -0.002851
```

The r values for each of the random graphs generated is negative. We can see that the Pearson correlation coefficient for the random networks indicates degree of disassortativity. Now that we have done the permutation test, it is possible to correctly conclude if the network is degree-assortative or disassortative based on the Pearson correlation coefficient of the human ppi network and the p value obtained. In this case the p value obtained from the code snippet below is 1.0 and since the pearson correlation coefficient is less than 0 we can conclude that the degree assortativity of the human ppi network is **degree disassortative.**

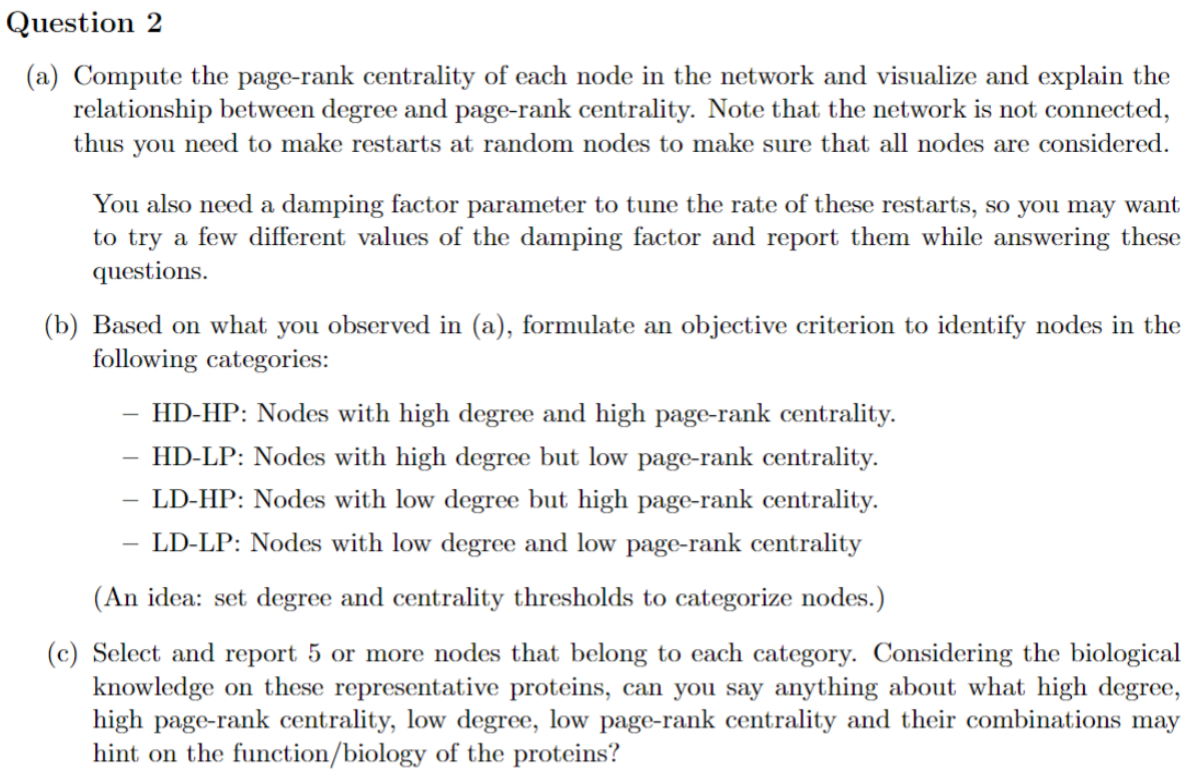
```
# Interpret the results based on the p-value and the
distribution of the random network's Pearson correlation
coefficients
if p_value < 0.05:
    if degree_assortativity > 0:
        print('The network is significantly
degree-assortative')
     else:
            print('The network is significantly
degree-disassortative')
```

```
else:
    print('The degree assortativity of the network is not significantly different from random networks')
```

Below is a visualization of the Pearson correlation coefficients calculated for the random networks as well that the value for the human ppi network as well.



Thus the p-value = 1.0, and the Pearson correlation for the human ppi network = `-0.05050037662470296` (indicated by the red line on the graph above) helps us conclude that the human ppi network is degree disassortative.



Question 2

(a) Compute the page-rank centrality of each node in the network and visualize and explain the relationship between degree and page-rank centrality. Note that the network is not connected, thus you need to make restarts at random nodes to make sure that all nodes are considered.

You also need a damping factor parameter to tune the rate of these restarts, so you may want to try a few different values of the damping factor and report them while answering these questions.

(b) Based on what you observed in (a), formulate an objective criterion to identify nodes in the following categories:

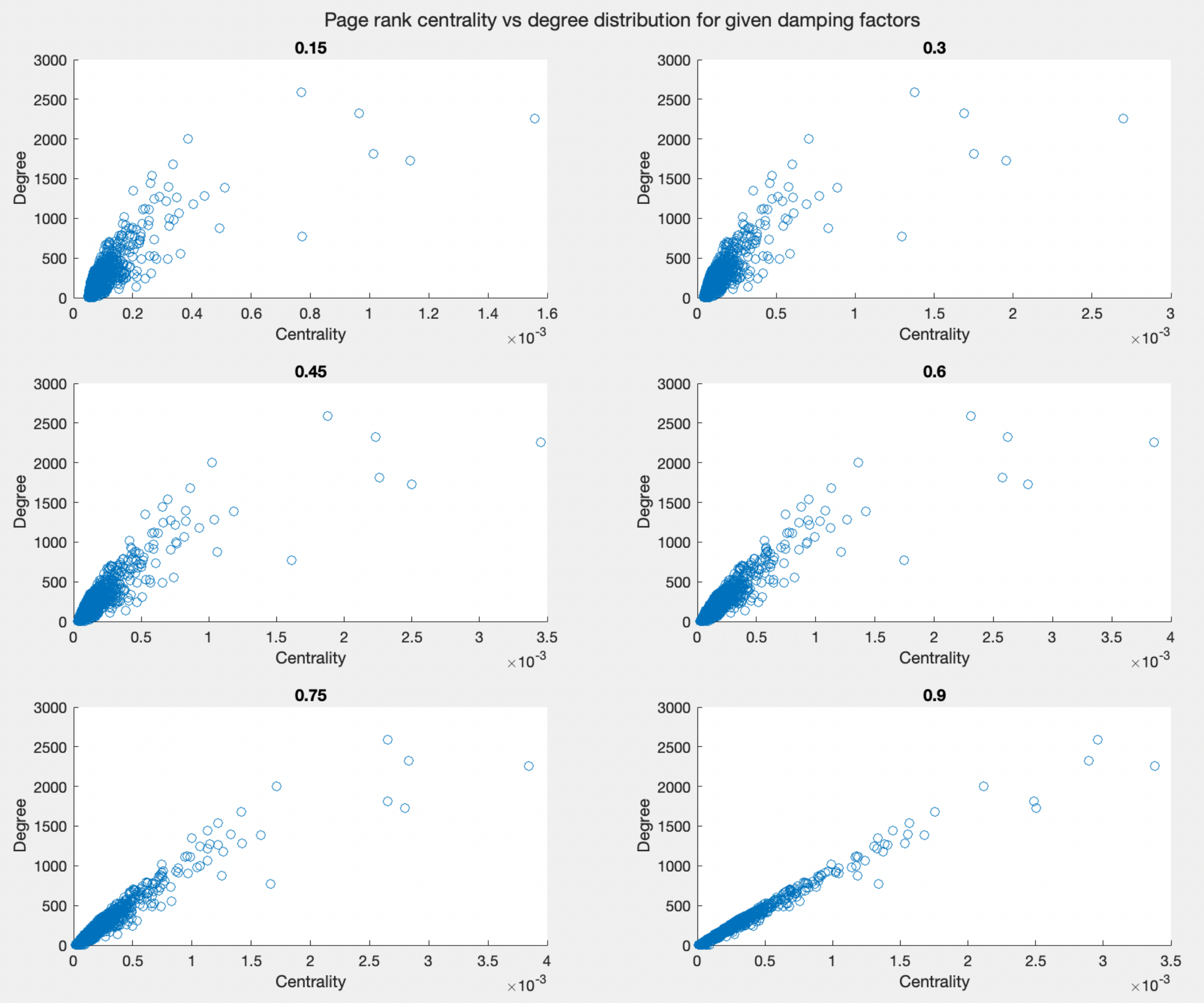
- HD-HP: Nodes with high degree and high page-rank centrality.
- HD-LP: Nodes with high degree but low page-rank centrality.
- LD-HP: Nodes with low degree but high page-rank centrality.
- LD-LP: Nodes with low degree and low page-rank centrality

(An idea: set degree and centrality thresholds to categorize nodes.)

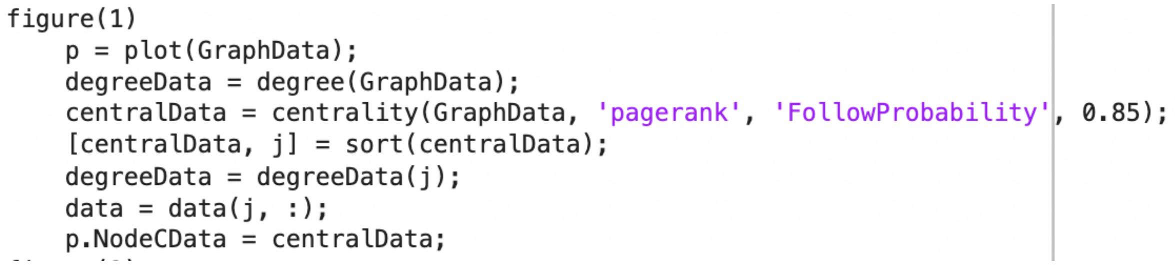
(c) Select and report 5 or more nodes that belong to each category. Considering the biological knowledge on these representative proteins, can you say anything about what high degree, high page-rank centrality, low degree, low page-rank centrality and their combinations may hint on the function/biology of the proteins?

Question 2

a. From the 6 graphs below that we got we see that there is a strong positive relationship between the degree centrality and the pagerank centrality. This would suggest that highly connected proteins are also highly important. This indicates that the network has a hierarchical structure in which highly connected proteins are critical for maintaining the network's overall structure and function. However, they are still proteins downstream who perhaps only have specific functions. In addition the damping factor decides if the algorithm jumps the nodes depending on the probability of the damping factor which can be a value between 0 and 1. The lower the damping factor, the more skewed the distribution will be because there will be more frequent jumps to another node, whereas a higher damping factor will indicate a more uniformed distribution because the algorithm follows a single chain of nodes before jumping randomly to another node and starting again. This can be seen that as the damping factor increases, the curve formed by the points is more uniform. However, it can also be seen that 0.75 is most likely the optimal damping factor for this network.

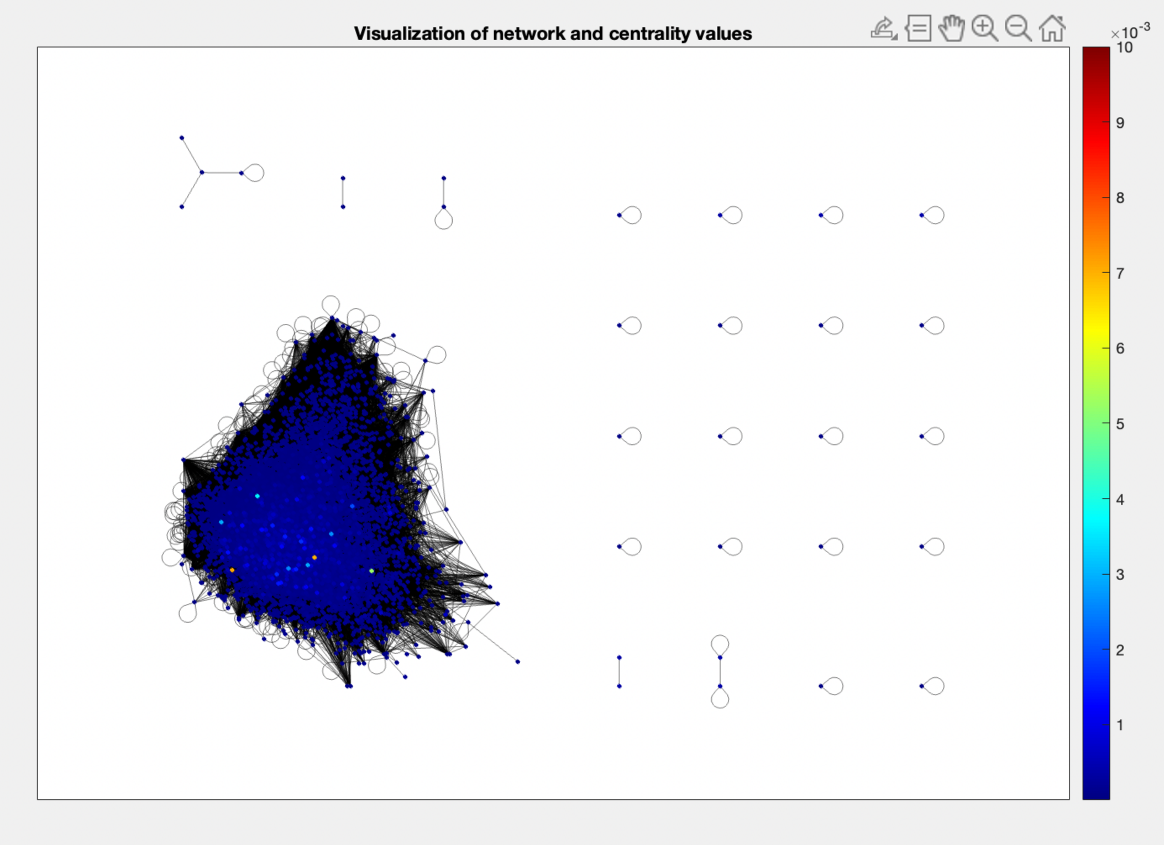


The code for calculating the centrality can be seen below. The function ‘graph’ and ‘centrality’ from Matlab were used.

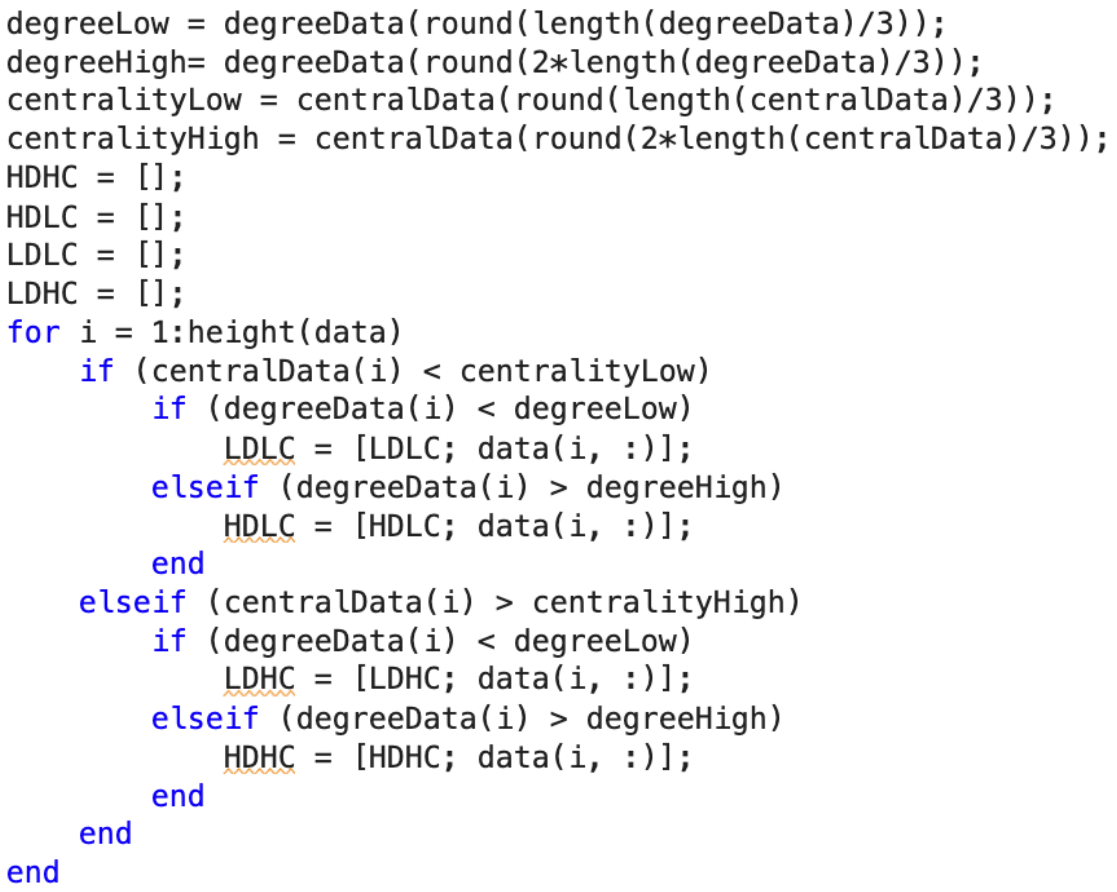


```
figure(1)
    p = plot(GraphData);
    degreeData = degree(GraphData);
    centralData = centrality(GraphData, 'pagerank', 'FollowProbability', 0.85);
    [centralData, j] = sort(centralData);
    degreeData = degreeData(j);
    data = data(j, :);
    p.NodeCData = centralData;
```

You can also see the network below, visualized with the centrality values.



b. We tried two different objective criteria. For the first, we said that proteins have high centrality if they are in the top 33%, and they have low if they are in the bottom 33%. The same was done for the degree distribution. Then, using above thresholds, we determine nodes in each criteria. However, it was realized that they are no nodes that have a high degree and low centrality. Therefore, we also tried to split nodes in the middle. Therefore, any nodes in the top 50% are high and the others are low. The code for finding the first criterion can be seen below.



```
degreeLow = degreeData(round(length(degreeData)/3));
degreeHigh= degreeData(round(2*length(degreeData)/3));
centralityLow = centralData(round(length(centralData)/3));
centralityHigh = centralData(round(2*length(centralData)/3));
HDHC = [];
HDLC = [];
LDLC = [];
LDHC = [];
for i = 1:height(data)
    if (centralData(i) < centralityLow)
        if (degreeData(i) < degreeLow)
            LDLC = [LDLC; data(i, :)];
        elseif (degreeData(i) > degreeHigh)
            HDLC = [HDLC; data(i, :)];
        end
    elseif (centralData(i) > centralityHigh)
        if (degreeData(i) < degreeLow)
            LDHC = [LDHC; data(i, :)];
        elseif (degreeData(i) > degreeHigh)
            HDHC = [HDHC; data(i, :)];
        end
    end
end
```

c. These are the nodes that belong to each category (for both criteria)

	High (top 33%) and low (bottom 33%)	High (top 50%) and low (bottom 50%)
HD-HP	'BAZ1A'- 'SATB1' 'BAD' - '1433Z' 'PP1A' - 'IPP2' 'CD70' - 'CD27' 'NCALD'- 'TBA4A'	'MYOTI' - 'MYOTI' 'IPO13' - 'MGN' 'AURKB' - 'H31' 'ITCH’ - 'ITCH' 'ARI3A' - 'P53'
HD-LP	-	'RPH3L' - 'RB27A' 'ELP1' - 'IKKB' 'PHC1' - 'SCMH1' 'RFA3' - 'RFA1' 'PPCE' - 'TKN1'
LD-HP	'MYH1' - 'STAU1' 'IMB1' - 'RBP2' 'DYL1' - 'K6PF,PFKAM' 'FHL2' - 'K6PF,PFKAM'	'ATS4' - 'ATS4' 'COIA1' - 'CATL1' 'SETD7' - 'HCFC1' 'TGFB1' - 'SPRC'

	'KCC2G' - 'FLNA'	'ATM' - 'RAD9A'
LD-LP	'ODP2' - 'PDK1' 'ADRB1' - 'GRB2' 'RPB3' - 'RPB4' 'JAM1' - 'AFAD' 'LOX12' - 'K2C5'	'ODP2' - 'PDK1' 'ADRB1' - 'GRB2' 'RPB3' - 'RPB4' 'JAM1' - 'AFAD' 'LOX12' - 'K2C5'

i. Nodes that have a high degree and high centrality are those proteins that have interactions with many other proteins, and can influence interactions with other proteins as well. They are hub proteins. For example, not only do they interact with many other proteins, but the proteins they interact with also have a high degree. These proteins would be ones that are upstream in pathways and have many interactions.

ii. Nodes that have a low degree and low centrality are those proteins that have interactions with very proteins and are not connected to many either. These proteins are probably going to be downstream in pathways, and have very specific functions.

iii. Nodes that have a low degree and high centrality are those proteins that are going to be connected to a few essential proteins. Therefore, while they can influence many other proteins causing them to have a high centrality value, they themselves have a low degree.

iv. Nodes that have a high degree but low centrality might be proteins that are part of a different pathway. Therefore, while they might be connected to quite a few other proteins, the centrality value is low as the 2nd pathway is completely disconnected from all the proteins in the first.