Lab class

1 - Preliminaries

Open an empty programming interface of your choice (e.g., Jupyter notebooks in Python). Download data for at least two relatively small unweighted and undirected networks[1] of somewhat different sizes from http://konect.cc/ and have a look at the data format. Import the network data into your interface, and visualize the network & corresponding adjacency matrix.

2 - Basic structural properties of networks

2a Focus on one of the above empirical networks. Compute the degree distribution and visualize its histogram. What is the mean and variance of the degrees?

2b Compute the local clustering coefficient, closeness centrality, and betweenness centrality of nodes in the network. Measure the correlation between degrees, betweenness, and closeness centrality. The correlation can either be estimated with the centrality values (Pearson) or with their associated ranking (Kendall's tau or Spearman's rho).

2c Create two visualizations for the empirical networks (i) nodes are colored based on degree and (ii) nodes are colored based on betweenness centrality.

2d Find the shortest path between all pairs of nodes. What algorithm is the built-in function using? Now, compute the number of components in the network, delete the 5% of nodes with highest betweenness centrality, and recompute the number of components. Has it changed?

2e Numerically compute the eigenvalues of the adjacency matrix, the (combinatorial) Laplacian matrix and the normalised Laplacian matrix of the empirical network. Verify the relationship between the number of components and the number of 0 eigenvalues of the Laplacian. What can be said about the range of the eigenvalues of the normalised Laplacian matrix?

3 - Random graph models

3a Consider an unweighted and undirected empirical network G. Generate (i) an Erdös-Rényi graph with an expected total edge weight equal to the total edge weight in G and (ii) a configuration random graph with the same degree sequence as G.

3b Sample networks (e.g., sample degree sequence from a power-law distribution) of increasing node set size from the configuration model. Compute the number of self-edges and multiedges for the different samples.

3c Generate a network using the BA model. Plot the resulting degree distribution and visualize the resulting network with nodes coloured by their degree (as in 2c).

4 - Community structure [Optional]

A *partition* of a network is a division of nodes into sets. For example $\{\{1,2,3\},\{4,5\}\}$ is a partition of 5 nodes into two sets.

4a Consider the matrix given by $\boldsymbol{B} = \boldsymbol{A} - (\boldsymbol{k}\boldsymbol{k}^T)/2M$, where $\boldsymbol{k}$ is an $N \times 1$ vector with i^{th} entry k_i. This matrix is often referred to as the *modularity matrix* and forms the basis for a very popular clustering method in network science known as *modularity maximization.* Import the Karate Club network from Konect and compute its modularity matrix.

4b Compute the leading eigenvector $\boldsymbol{v}_1$ of the modularity matrix of the Karate Club and assign nodes to two sets (thus obtaining what is known as a *bipartition*) as follows:

$$\text{node } i \text{ is in set 1 if } v_{1i} \geq 0 \text{ and node } i \text{ is in set 2 otherwise}\,,$$

with v_{1i} the i^{th} entry of $\boldsymbol{v}_1$. The assignment of nodes to communities can be encoded in an N-dimensional vector with i^{th} entry equal to 1 if node i is in set 1 and -1 otherwise. Write a function that takes as input a modularity matrix and a bipartition, and returns the following quantity as an output:

$$\sum_{\substack{i,j \\ \text{in same set}}} B_{ij}\,.$$

Compare the value of the above quantity for the partition as defined in 4b, and a partition where nodes are assigned to two sets uniformly at random.

4d Visualize the Karate Club network with nodes coloured based on their set assignment as obtained in 4b.

[1] If the empirical network of interest is weighted and directed, consider an unweighted (e.g., an edge is present if its weight is nonzero and absent otherwise) and undirected (e.g., assume every edge is reciprocated) version of it.