Homework 09 - Network Science

Gabriel Luciano Gomes (265673)

December 4, 2020

1 Computing degree correlation measures

1.1 Degree Correlation Matrix

This heatmap as based on the values of e_{ij} (degree correlation matrix), which indicates the probability of obtaining a node of degree j at the end of node i, on a random link selection. The e_{ij} values can be computted with :

$$\sum_{i,j} e_{ij} = 1$$

$$q_k = \frac{kp_k}{\langle k \rangle} \text{ (Deriving the probability } q_k\text{)}$$

$$\sum_{j} e_{ij} = q_i \text{ (Connecting } q_k \text{ to } e_{ij}\text{)}$$

Listing 1: Generating Correlation Matrix

```
import networkx as nx
import numpy as np

degrees = [y for x,y in G.degree()]
correlation_matrix = [[0 for i in range(max(degrees)+1)] for j in range(max(degrees)+1)]

# counting links from degree i to degree j
for (node_i, node_j) in G.edges:
    correlation_matrix[G.degree(node_i)][G.degree(node_j)] += 1
    correlation_matrix[G.degree(node_j)][G.degree(node_i)] += 1

correlation_matrix = correlation_matrix/np.matrix(correlation_matrix).sum()
```

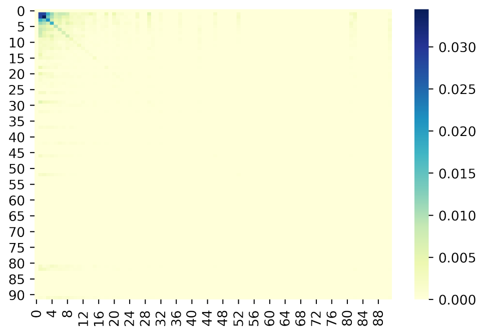

Figure 1: Correlation matrix plotted as heatmap

After plotting the heatmap, we can analise that low-degree nodes are linked to low-degree nodes. This characterize a **disassortative** network.

1.2 Degree Correlation Function

The degree correlation function captures the probability that a link of a k-degree node reaches a k'-degree node. Whereas can be computed by:

$$k_{nn}(k_i) = \frac{1}{k_i} \sum_{j=1}^{N} A_{ij} k_j$$

Listing 2: Computing the degree correlation function

```
import networkx as nx

degreeCorrelation = nx.average_degree_connectivity(G)
```

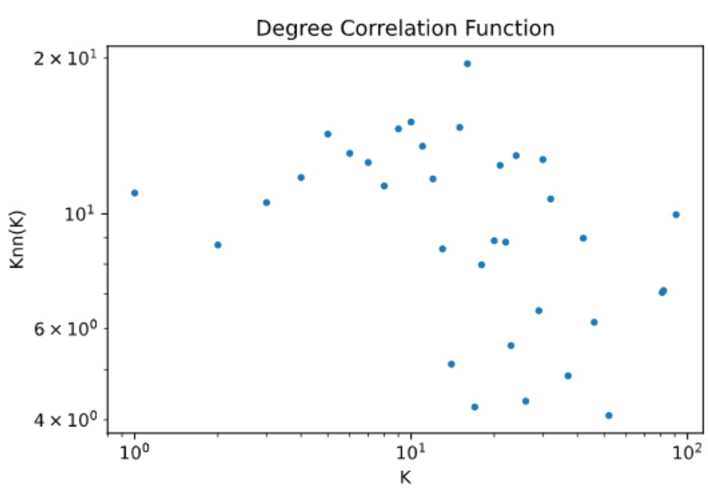

Figure 2: Degree correlation function

Analysing the degree correlation function, we can not identify a pattern of links. Low degrees nodes and also high degree nodes present a varied K_{nn}. Thus, the hubs do not have a prefference whether to link to low or high degree nodes. Finally, it denotes a **neutral** network.

1.3 Degree Correlation Foefficient

The degree correlation coefficient can be computed by:

$$r = \sum_{jk} \frac{jk(e_{jk} - q_j q_k)}{\sigma^2}$$

where

$$\sigma^2 = \sum_k k^2 q_k - \left[\sum_k k q_k \right]^2$$

Additionaly, the value of r varies in the interval $-1 \leq r \leq 1$, and with the representations: If $r < 0$, it means that it is a **disassortative** network; If $r = 0$, it means that it is a **neutral** network; If $r > 0$, it means that it is a **assortative** network.

Listing 3: Computing the degree correlation coefficient

```
import networkx as nx
nx.degree_pearson_correlation_coefficient(G)
```

For this network, the obtained degree correlation coefficient was $r = -0.0550781093422519$. As $r < 0$, it is a **disassortative** as visualized at figure 1. However, as the r is close to zero, we can observe at figure 2 that the network tends to be neutral.