BST 267: Introduction to Social and Biological Networks

Lab 4

JP Onnela

Department of Biostatistics
Harvard T.H. Chan School of Public Health
Harvard University

November 17 & 18, 2022

Modeling Network Structure

- The goal of this lab is to learn how to generate ER random graphs and to learn about their phase transition
- **Deliverable:** Return these through Canvas: 1) Jupyter Notebook (.ipynb file) and 2) HTML version of the notebook (.html file)

Question 1

- We will generate $G(N, p)$ graphs without using the `nx.erdos_renyi_graph` function
- Write code that generates a graph from the $G(N, p)$ ensemble (the classic Erdős-Rényi graph)
- The code will need two parameters: the number of nodes N and the connection probability p
- Function `random.random` generates a random number in the $[0, 1)$ interval
- We can use it to simulate the outcome of a Bernoulli variable:

```
import random
if random.random() < 0.5:
    print("Heads")
else:
    print("Tails")
```

- An alternative to `random.random` is `scipy.stats.bernoulli.rvs`:

```
from scipy.stats import bernoulli
bernoulli.rvs(p=0.2)
bernoulli.rvs(p=0.2, size=20)
```

- Use the following parameters: $N = 100, p = 0.05$

Question 2

- The easiest way to generate Erdős-Rényi graphs in NetworkX is using the `nx.erdos_renyi_graph` function:

```
import matplotlib.pyplot as plt
%matplotlib notebook
G = nx.erdos_renyi_graph(20, 0.15, seed=123)
nx.draw(G)
```

- Erdős-Rényi networks show a percolation (phase) transition when the link probability (link density) $p \approx 1/N$ where N is the number of nodes in the network
- Since average degree in Erdős-Rényi networks is given by $\langle k \rangle = p(N-1) \approx pN$, the transition happens when $\langle k \rangle = 1$

Question 2

- Study this transition by making a plot of the relative size of the largest connected component $R_{LCC}(p)$ as a function of p around $p \approx 1/N$
- You can use a small network to do this (e.g., $N = 50$)
- Instead of investigating one realization for each value of p, calculate the average value of $R_{LCC}(p)$ over 10 realizations and plot that against p
- If your network is stored in `G`, you can extract the largest connected component of `G` using the following:

```
Gc = max(nx.connected_component_subgraphs(G), key=len)
```

Question 2

- You can import the NumPy module using `import numpy as np`
- You can calculate the mean of a sequence using the `np.mean` function
- You can create a semi-logarithmic plot using the `plt.semilogx` function
- You can create a NumPy array (a type of sequence) of evenly spaced p values using `ps = np.logspace(-np.log10(N)-1, -np.log10(N)+1, 40)`