

Exercise set #2 (15 pts)

- The deadline for handing in your solutions is October 1st 2018 23:55.
- Return your solutions (one .pdf file and one .zip file containing Python code) in MyCourses (Assignments tab). Additionally, submit your pdf file also to the Turnitin plagiarism checker in MyCourses.
- Check also the course practicalities page in MyCourses for more details on writing your report.

1. Ensemble averages by enumeration (3 pts, pen and paper)

Graph *ensembles* are distributions of graphs, where each graph G_i has a certain probability π_i . The *ensemble average* of a quantity X is defined as $\langle X \rangle = \sum_i \pi_i X(G_i)$. Note that this is the expected value of X across the ensemble. Let us define following quantities: $\langle k \rangle$ is the average degree of the graph G , $\langle c \rangle$ is the average clustering coefficient for graph G (assuming that the clustering coefficient gets value 0 for nodes of degree 0 and 1), and $d^*(G)$ is the diameter of the connected component of the graph G that has the largest diameter.

- (2 pt) **Calculate**, using pen and paper, $\langle k \rangle$, $\langle c \rangle$, and $\langle d^* \rangle$ for $G(N=3, p=1/3)$ in which N is the number of nodes and p is the edge density.
- (1 pt) **Calculate**, using pen and paper, the formulas for $\langle k \rangle$, $\langle c \rangle$, and $\langle d^* \rangle$ for $G(N=3, p)$ (that is, for $N=3$ and all possible values of p). Remember to simplify the formulas you get as results.

Note that there are in general $2^{N(N-1)/2}$ possible networks with N nodes! That is, for $N=3, 4, 5, 6, \dots$ there are 8, 64, 1024, 32768, ... possible networks, and for $N=100$ the number of possible networks is in the order of 10^{1490} . The naive method used in this exercise for calculating ensemble averages directly from the definition quickly becomes unpractical when the number of nodes increases. In the exercise 3 you will use a more practical method for estimating ensemble averages.

2. Properties of Erdős-Rényi (ER) networks (5 pts)

Erdős-Rényi networks are random networks where N nodes are randomly connected such that the probability that a pair of nodes are linked is p . In network science, the ER random graphs are important because they provide the simplest reference to which one can compare real-world networks. In this exercise, we will analyze some of the properties of ER graphs.

- (1 pt) The degree distribution $P(k)$ of ER networks is binomial:

$$P(k) = \binom{N-1}{k} p^k (1-p)^{(N-1)-k}, \quad (1)$$

where N equals the number of nodes in the network. **Explain in detail** the origin of each of the three factors in the above formula. You don't need to derive the distribution, but it should be visible from your answer that you understand the meaning of each of the terms.

- b) (1 pt) In ER networks, the expected value of the average clustering coefficient $\langle c \rangle$ equals p . **Explain**, why this is the case.

Hints:

- What is the expected value of the clustering coefficient for one node?
- **Note:** You don't need to provide a detailed mathematical proof. Some clear general reasoning is enough.

- c) (1 pt) **Explain**, what happens to $\langle c \rangle$, if $N \rightarrow \infty$ with $\langle k \rangle$ bounded.

Hint:

- Mathematically, if x is 'bounded' then x is always smaller than some possibly big, but finite, number M .

- d) (2 pts) Use NetworkX to calculate estimates for the ensemble averages $\langle k \rangle$, $\langle c \rangle$, and $\langle d^* \rangle$ defined in the Exercise 1. Do this by generating 100 ER networks for each value of p in range $p = [0.00, 0.05, \dots, 0.95, 1]$ and $N = 3$. An estimate for the ensemble average $\langle X \rangle$ can be calculated for each value of p by calculating the average value of X over the 100 realisations. **Plot** the quantities, such that p values are on the x-axis and $\langle X \rangle$ values are on the y-axis. If you solved the part b of Exercise 1, you can check the correctness of your results by including your analytical solution to these plots. Repeat the same exercise with $N = 100$.

Hints:

- You can use the function `nx.fast_gnp_random_graph(N, p)` for generating ER networks.
- You can use the function `nx.diameter()` to calculate the diameter of a network. However, this function only accepts graphs that have a single connected component. Use the function `nx.connected_component_subgraphs()` to iterate over the components, calculate the diameter for each of them, and find the largest value d^* .
- There is a python template code provided for this exercise (`properties_of_erdos_renyi.py`). However you can also write your own code from scratch if you want.

3. Implementing the Watts-Strogatz small-world model (7 pts)

In this exercise, you will implement the Watts-Strogatz small-world model [1], which is a very simple network model that yields small diameter as well as high level of clustering. In practice, the Watts-Strogatz model is a ring lattice where some of the links have been randomly rewired. The model has three parameters: network size N , m (each node on the ring connects to m nearest neighbors both to the left and to the right), and p , the probability of rewiring one end of each link to a random endpoint node.

- a) (4 pts) **Implement** the Watts-Strogatz small world model and **visualize** the network using $N = 15$, $m = 2$ $p = 0.1$, and $N = 100$, $m = 2$ $p = 0.05$ using a circular layout algorithm (`nx.draw_circular(G)`), and check that the networks look right. For each network **Report** the total number of links and also the number of rewired links.

Hints:

- See the code template `implementing_ws_model.py`

Note: NetworkX has a ready-made function for small-world networks but here the task is to program your own function, so do not use it (except for checking results, if in doubt).

- b) (3 pts) **Plot** the *relative* average clustering coefficient $c(p)/c(p=0)$ and average shortest path length $l(p)/l(p=0)$ vs. p in your network, for $p = 0.001, \dots, 1.0$ (see template for the log-spaced values). Here, relative=average value for given p divided by the same value for $p=0$. **Use** $N = 500$ and $m = 2$. Use a logarithmic x-axis in your plot (`ax.semilogx`). **Check** that your results are in line with the plots in the lecture slides.

Hints:

- See the code template (`implementing_ws.py`).

Feedback (1 pt)

To earn one bonus point, give feedback on this exercise set and the corresponding lecture latest two day after the report's submission deadline.

Link to the feedback form: <https://goo.gl/forms/3KuMA7PG81wr2UJp2>.

References

- [1] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, pp. 440–442, 1998.