Assignment 2

Madeline Schiappa

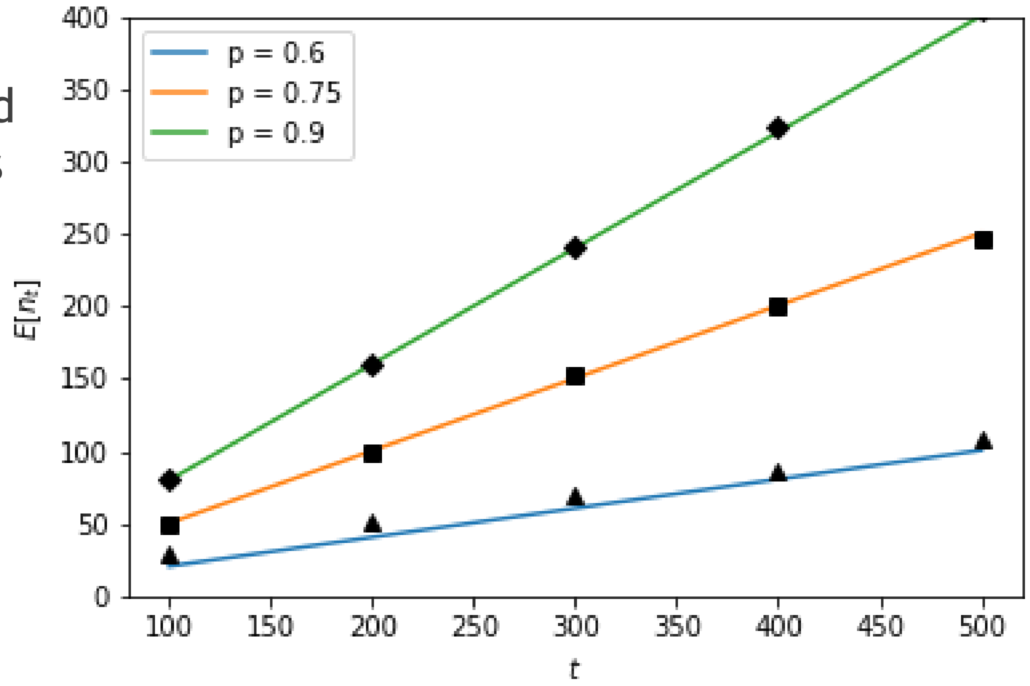
Analytical solutions by Narsingh Deo and Aurel Cami (2017)

Simulation

- The simulation iterates through each probability of a node and edge birth, through the number of simulations, and then the number of time steps.
- During each time step, it adds the number of nodes and edges to a set variable,
  - Once all the simulations complete for that probability, it divides it by the number of simulations in order to get the "expected value"
- The simulation also does this for .8 percent to calculate a simulated degree distribution and an expected degree distribution from an analytical solution.

Figure 2

- The shapes are the simulated expected value and the lines are the analytical expected value of nodes.

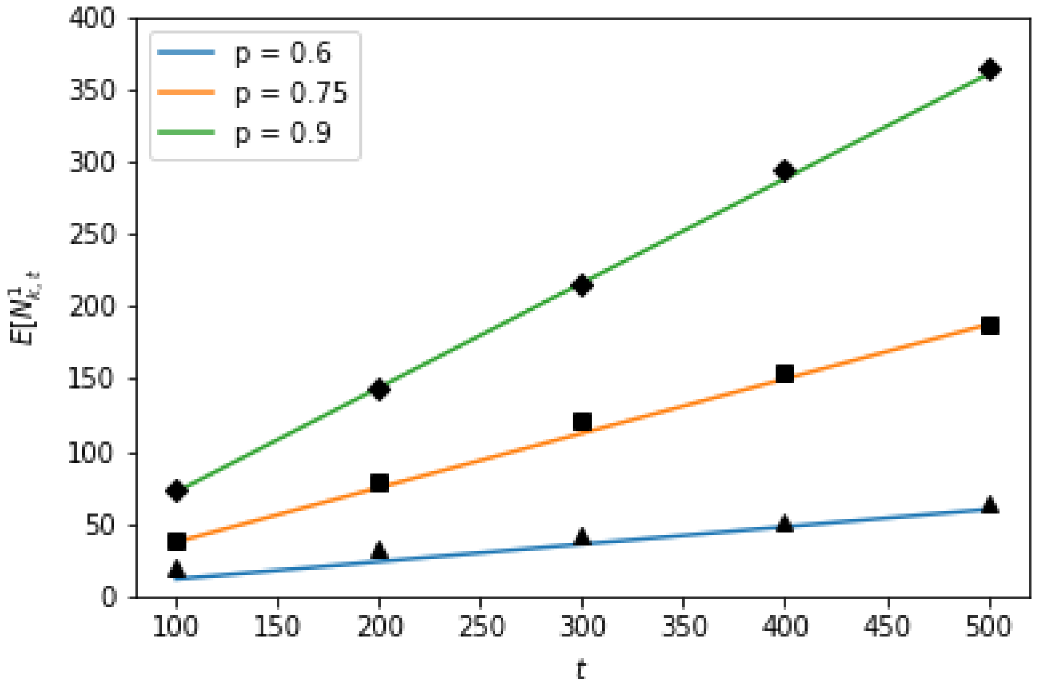



```
def expected_nodes(p, q, t):
    return (p - q) * t + 2 * q


def run_expected_nodes(p, time_steps):
    nodes = list()
    for t in range(time_steps):
        nodes.append(expected_nodes(p, 1 - p, t + 1))
    return nodes
```

Figure 3

- The shapes are the simulated expected value and the lines are the analytical expected value of edges.

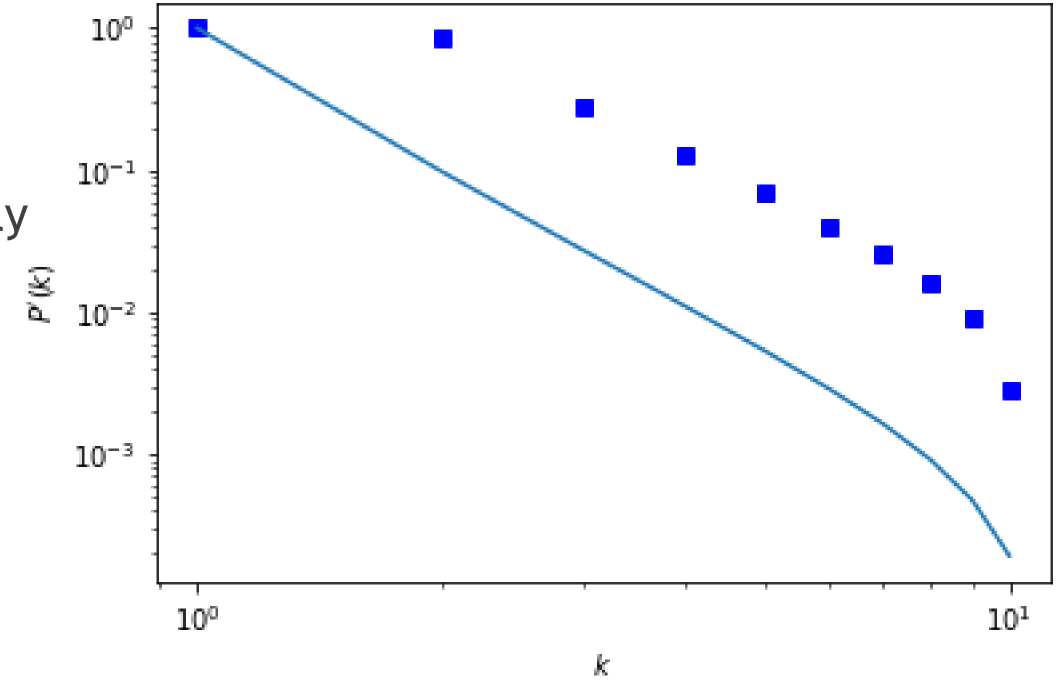



```
def expected_edges(p, q, t):
    # a = (2.0*q)/(p-q)
    e = (p * (p - q)) * t
    # return (1+a)*e
    return e


def run_expected_edges(p, time_steps):
    edges = list()
    for t in range(time_steps):
        edges.append(expected_edges(p, 1 - p, t + 1))
    return edges
```

Figure 5

- This compares the expected cumulative degree of the simulated and the analytically derived expected.




```
def expected_degree_dist(p, k):
    return k ** (-1 - ((2 * p) / (2 * p - 1)))
```