Alexander Looi
Homework 5
March 29th, 2017
Write up

1. I think the best data structure to store the community data would be a dual dictionary. In this data structure the key for one dictionary would be the community name, here values for each key would be nodes that belong to a community. The other dictionary would have keys that represent each node and the community or communities that node belongs to. This way we can either ask to retrieve all the nodes of a community or retrieve the community of a node by using memory instead of spending time build type from a single data structure.
2. I think the best way would be an edge list esque format. In the first column is a node, and the second column held the community that the node belonged to.

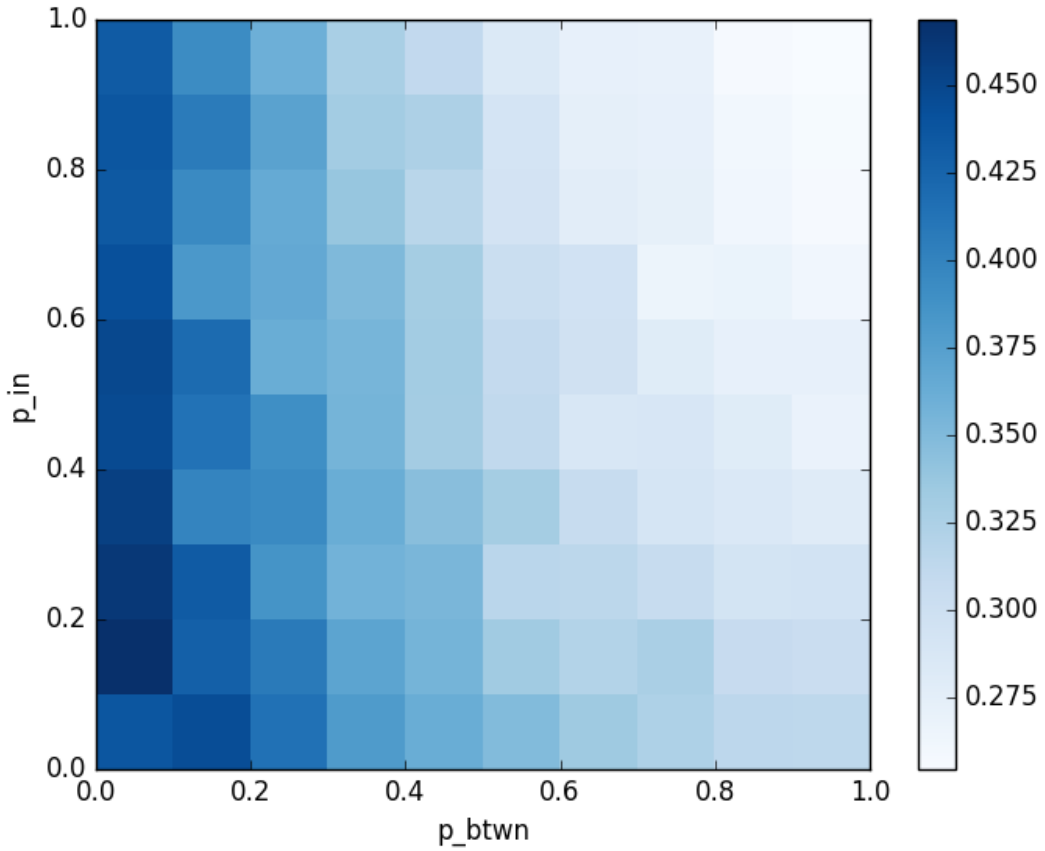

Figure 1. 4 community graph where each community has 5 nodes.

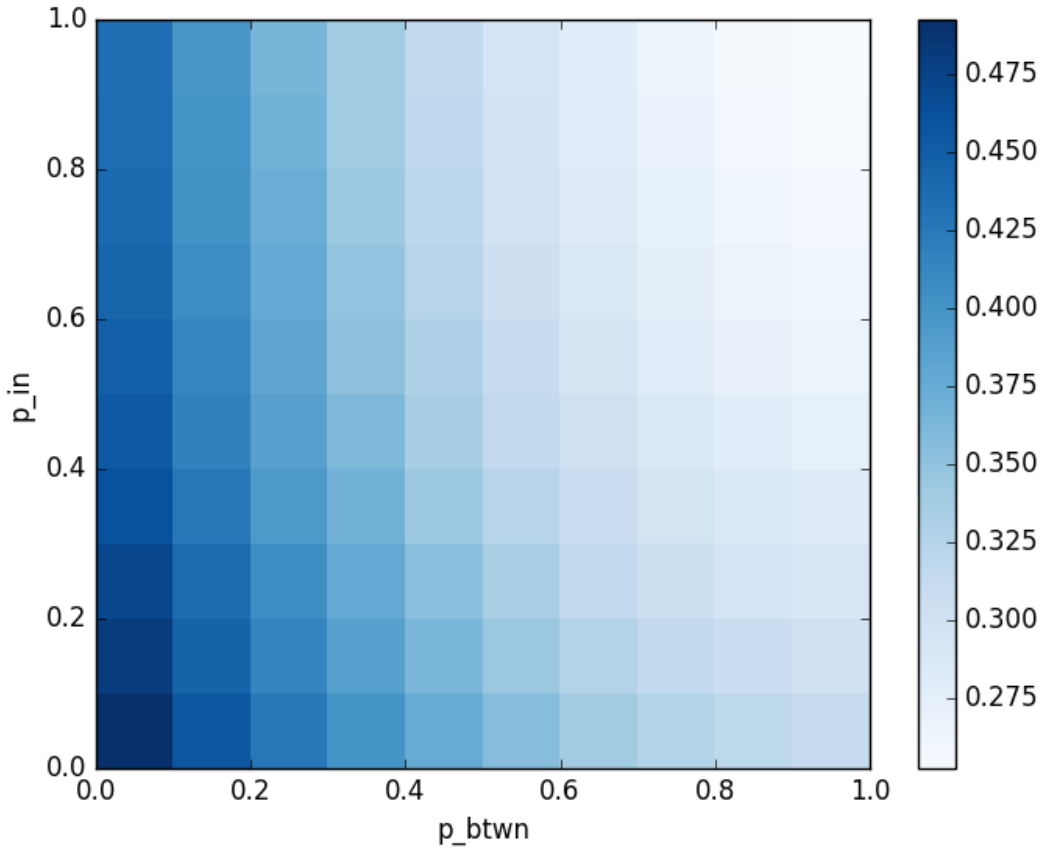

Figure 2. 4 community graph where each community has 50 nodes

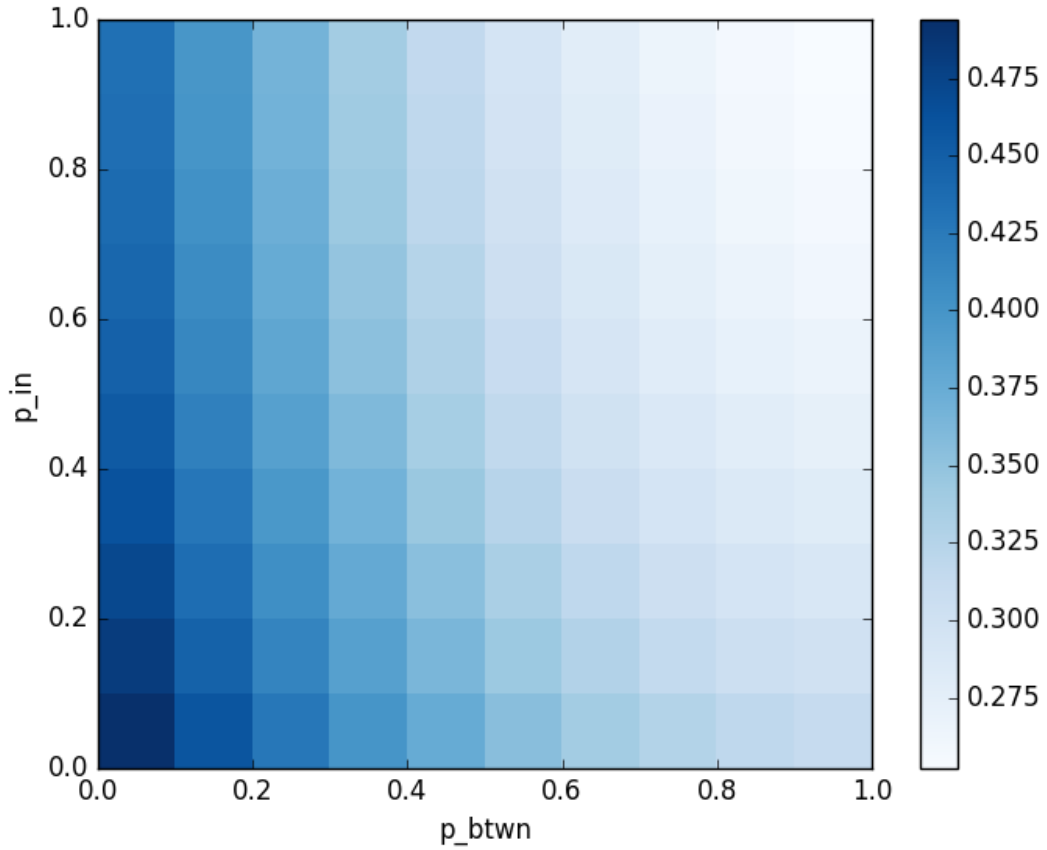

Figure 3. 4 community graph where each community has 75 nodes.

3. I think there is something wrong with my modularity calculation since the highest modularity values appear when there are no nodes connected between and within communities, particularly at high node per community values (50 and 75). The 5 node communities produced more interesting results. Here, the highest modularity values appeared between 0.1 and 0.2 prob of p_in with ~0.01 p_btwn. This could be an artifact of resolution, but again it's difficult to determine since my modularity values are "weird."
4. I'm going to attempt a greedy algorithm. Here we loop through all nodes and look at the nodes neighbors. If adding the node increase modularity then set those nodes as part of a community. I'm going to attempt another version that determines if a neighboring node is part of another community, if it is, include those nodes in the modularity calculation.