Greedy algorithm to perform community detection of networks following Dynamic Stochastic Block Model

barnab.nitrkl

October 2018

1 Algorithm Description

In order to find the community assignment of a particular node at time instant t, our algorithm will use the community assignment of that particular node at time instant t-1 and t+1 as well as its community assignment from the previous iteration to update the assignment for the current iteration. In order to find out the initial class assignment of the nodes in the graphs we perform spectral clustering. Spectral clustering will not consider the properties of dynamic stochastic block model and thus our approach will iteratively update the community assignment of nodes of a graph at time instant t with the help of community assignment of nodes of graph at time instant t-1,t and t+1.

Given a t-length sequence of DSBM networks the first step of the algorithm is to find the spectral clustering assignment of nodes of the entire sequence of graphs. We will define a 'Class Density' as the ratio as the number of edges shared by nodes belonging to a particular class assignment to the number of nodes in that class. Once spectral clustering is done, we can easily find the class densities of each class in the graphs.

Once this initial assignment is done, our algorithm will look to further smoothen the class assignment using the following logic:

Given a node n of a graph G at time instant t, three counts of edges needs to be maintained. The first count will count the number of nodes in G which share an edge with n and was assigned the community k in the previous timestamp. The second count will maintain the count of the number of edges in G which share an edge with n and was assigned the community the community k in the next timestamp. The third count will maintain the count of the number of nodes in G which share an edge with n and was assigned the community k in the current timestamp of the previous iteration.

We assume that every pair of node has exactly one edge and thus the above counts also correspond to the number of edges which node n share with its neighbours belonging to a community. Given the count of the edges, a high ratio of the count and the class density indicates that the node has higher chances of belonging to that particular class. Using this heuristic we find the most probable class for node n using the graph from timestamp t-1. We also find the most probable class for the node using the graph from timestamp t+1.

If the most probable class for both the timestamp is the same, the assignment of n in the graph Gt will be updated as the most probable class. Since the graphs evolve as a DSBM it looks logical that at time t the class of node n would also have been the same as the class assignment at time t-1 and t+1.

If the most probable class for both the timestamps are different, then we need to inspect those with the class assignment at the current timestamp t. If the current class assignment matches with most probable class for either t-1 timestamp or t+1 timestamp then we keep the current class assignment to maintain continuity. If the current class assignment does not match with both t-1 and t+1 most probable class then we choose the class assignment with the highest value of count/class density.