Project 1: Random Graphs and Random Walks

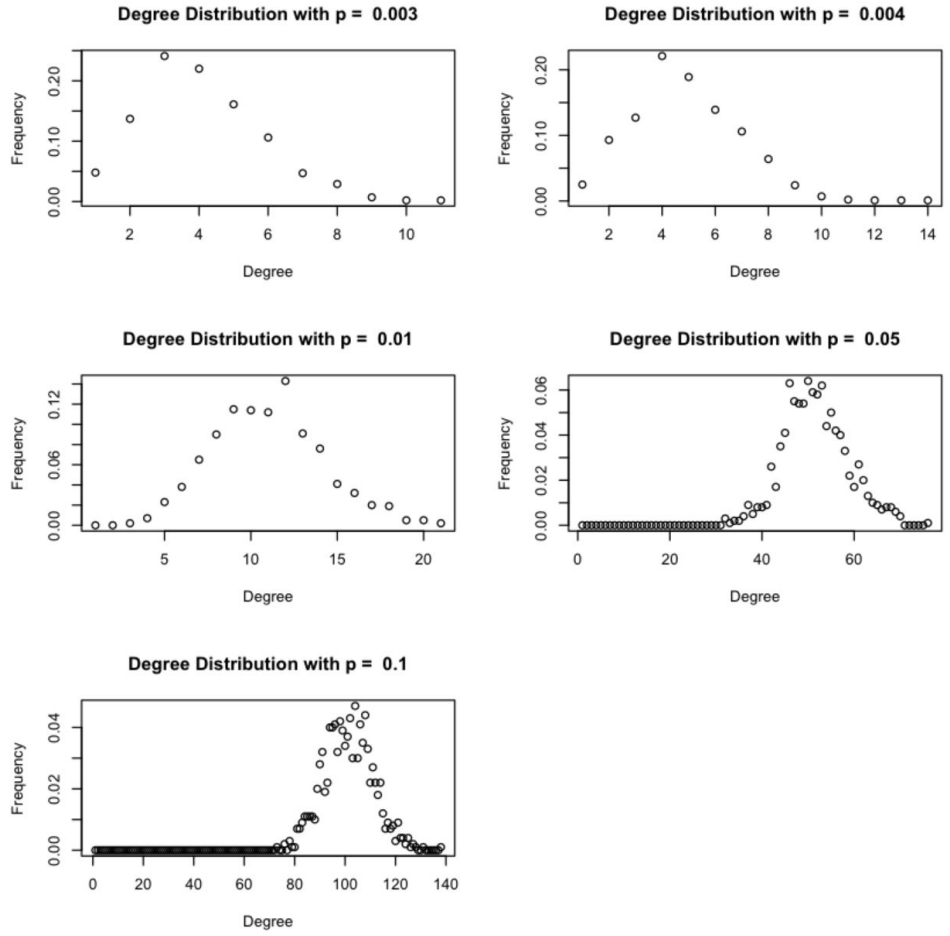
Duan Li 005026839
Di Ma 004945175
Yuan Shao 504880181
Weijie Tang 305029285

Part 1: Generate Random Networks

1. Create random networks using Erdös-Rényi (ER) model

(a)

Degree Distribution Graph



We use erdos.renyi.game to create the network and degree.distribution to calculate the degree distribution. From the degree distribution graph, we observe binomial distributions with different means.

$$\binom{n-1}{k} p^k (1-p)^{n-1-k}$$

where p is the probability that an edge is present, n is the number of nodes, which is 1000.
When p = 0.003, we observe the binomial distribution with mean 2.994.
When p = 0.004, we observe the binomial distribution with mean 4.050.
When p = 0.01, we observe the binomial distribution with mean 9.922.
When p = 0.05, we observe the binomial distribution with mean 49.832.
When p = 0.1, we observe the binomial distribution with mean 99.574.

We use E = (n-1) * p to calculate theoretical mean and Var = (n-1) * p * (1-p) to calculate theoretical variance. n-1 rather than n is used because the package doesn't create self-loops. The observed mean matches and the theoretical mean and thus the generated networks are indeed ER models.

	p = 0.003	p = 0.004	p = 0.01	p = 0.05	p = 0.1
mean	2.994	4.050	9.922	49.832	99.574
theoretical mean	2.997	3.996	9.990	49.950	99.900
variance	2.872837	3.879379	9.921838	45.231007	98.390915
theoretical variance	2.988	3.980	9.890	47.453	89.910

The mean of distribution increases as p increases. It is because larger p indicates higher chance of a node to be connected with more nodes. When p is relatively small, like 0.003, most nodes do not have edges and thus only have degree of 0. When p is relatively large, like 0.1, most nodes have many edges and thus have degree larger than 0. Thus, when p is increasing, more nodes will have a larger degree and the mean of the degree distribution will be increasing.

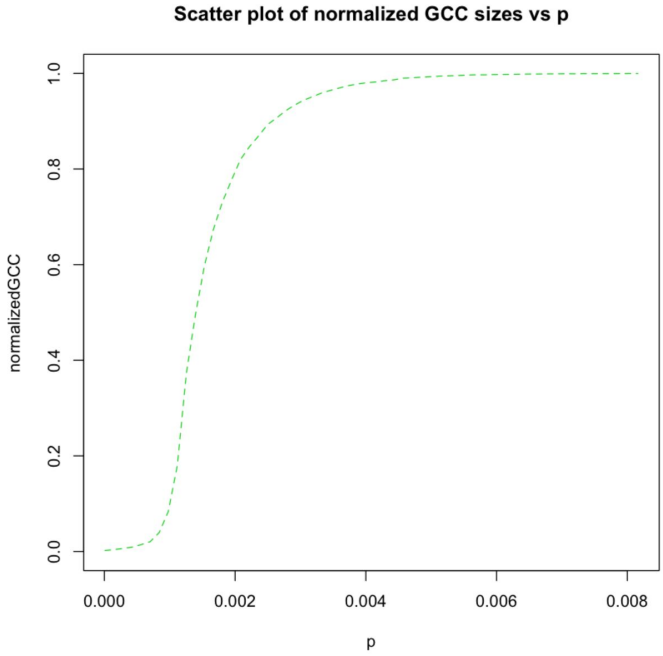
(b) Not all random realizations of the ER network is connected.
We iterate for 2000 times and use is.connected and diameter functions to check the connected probability and the GCC diameter. The table below shows results.

	p = 0.003	p = 0.004	p = 0.01	p = 0.05	p = 0.1
connected probability	0	0	0.9615	1	1
GCC diameter	17	10	5	3	3

For p = 0.003, the generated network has 0% connectivity and the diameter is 17.
For p = 0.004, the generated network has 0% connectivity and the diameter is 10.
For p = 0.01, the generated network has 96.15% connectivity and the diameter is 5.

For p = 0.05, the generated network has 100% connectivity and the diameter is 3.
For p = 0.1, the generated network has 100% connectivity and the diameter is 3.

(c) The graph below is the scatter plot of the normalized GCC sizes vs p.

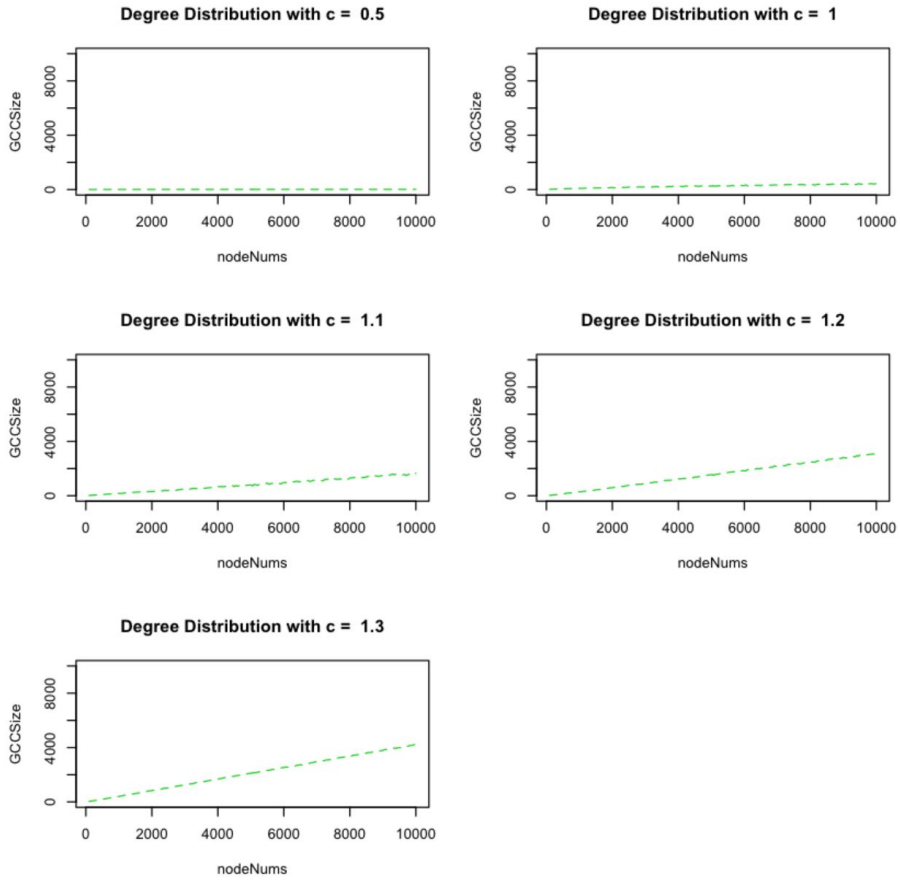


Let's define emergence as "a threshold where the network is not connected when p is below this threshold and the network is connected when p is above this threshold".

If np < 1, E(|GCC|) = O(ln(n)). So if p < 0.001, normalized GCC size is almost 0.
If np = 1 E(|GCC|) = O(sqrt(n)) So if p = 0.001, normalized GCC size = sqrt(1000)/1000 = 0.031.
If np > 1 E(|GCC|) = constant * n. So if p > 0.001, normalized GCC size is proportional to n before emergence.

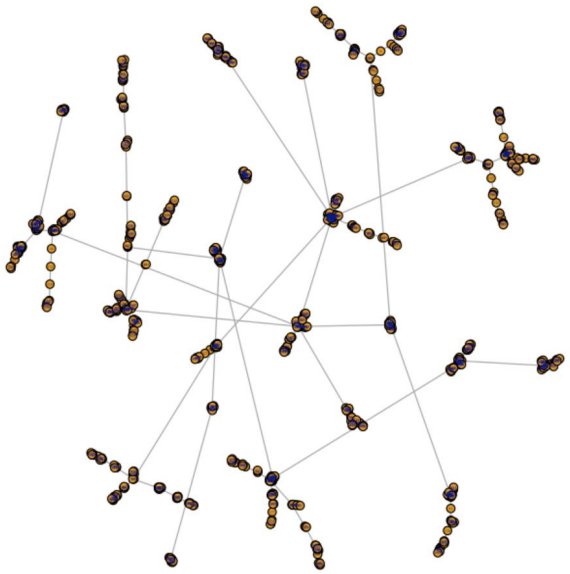
Based on our model, the estimated p where a giant connect component starts to emerge is 0.006. The theoretical value of p is calculated by O(ln(n)/n) = 0.0069. So the estimated p matches the theoretical value.

(d) Below is the plot of the expected size of the GCC size when c = 0.5, 1, 1.1, 1.2, and 1.3. For c = 0.5, the plot is almost a horizontal line. For c = 1.3, the plot has stiffest slope. The trend observed is that the slope of the plot is increasing when c is increasing. So we know that the GCC size increases as the average degree of nodes increase. In other words, GCC size is proportional to the average degree of nodes.



2. Create networks using preferential attachment model

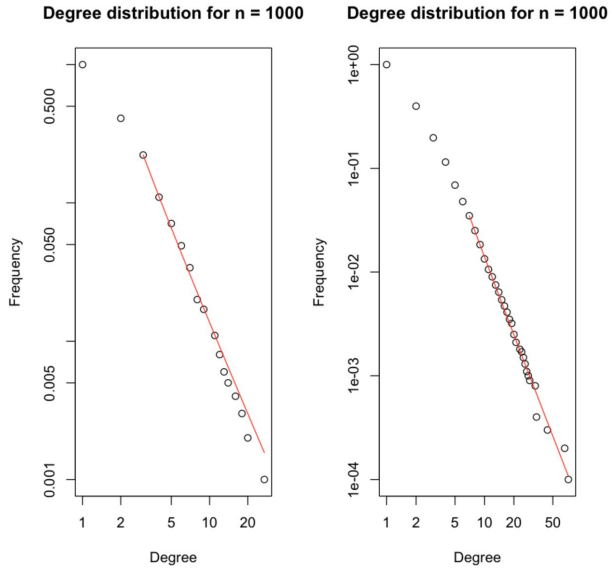
(a) Yes, a network with n = 1000 nodes and each new node attaching to m = 1 old nodes is always connected. We use barabasi.game to generate the network as below.



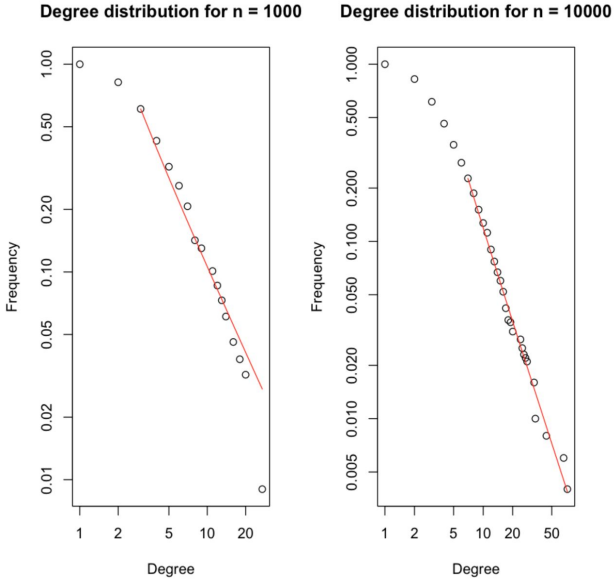
(b) The modularity of network using fast greedy method (fastgreedy.community) is 0.933.

(c) The modularity of network with 10000 nodes is 0.978, which is larger than the modularity of network with 1000 nodes.
Modularity measures the strength of division of a network into modules (clusters, communities, or groups). In this case, the larger network with 10000 nodes have denser connections between the nodes within modules but sparser connections between nodes in different modules than the smaller network with 1000 nodes. In other words, the larger network is more easily to be divided into communities than the smaller network.
Therefore, larger network has smaller edge density and larger modularity than smaller network.

(d) The estimated slope of degree distribution is -3.116 for n = 1000, -3.414 for n = 10000.

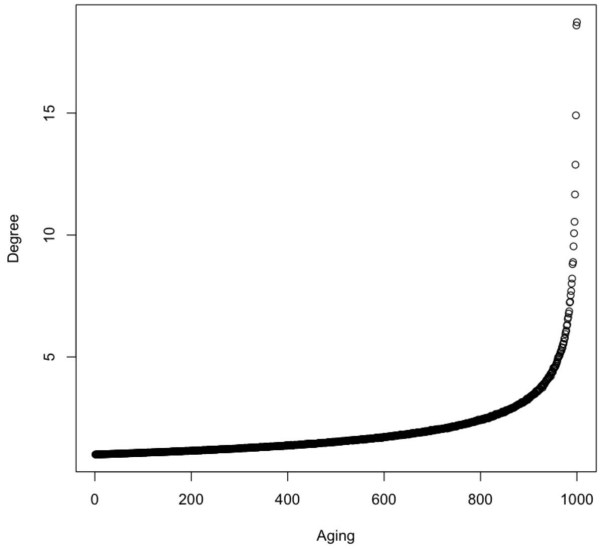


(e) The estimated slope of node j deg distribution is -2.323 for n = 1000, -2.699 for n = 10000.

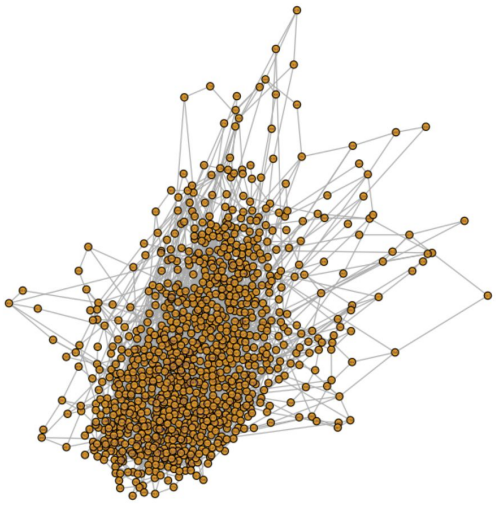


The slope of plot in part (d) is slightly larger and stiffer than the slope of plot in part (e). The plot in part (d) is the degree distribution of preferential attachment model and the plot in part (e) is the degree distribution of uniform picked node at random. Preferential attachment model prefers to link a newly added node to a node that already has high degree. So degrees of older nodes in (d) is greater than that in (e), which results in “heavy tail” and relatively stiff slope in (d). Similarly, degrees of younger nodes in (d) is smaller than that in (e), which leads to a flat tail and relatively less stiff slope in (e).

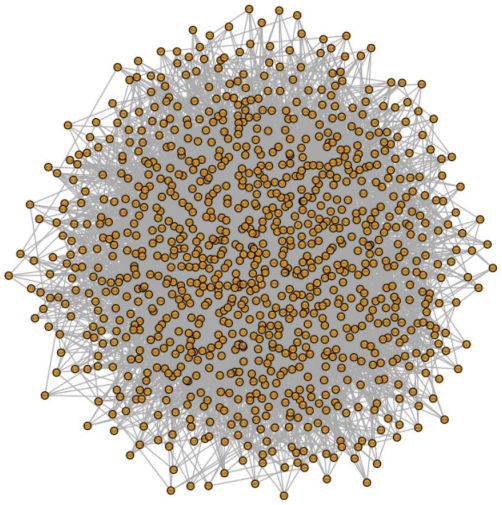
(f) From the plot, we know that nodes with higher degree have stronger ability to grab links added to the network. In other words, older nodes have higher degrees.



(g) The network with n = 1000 nodes and each new node attaching to m = 2 or m = 5 old nodes is always connected. The plots are shown as below.



m=2

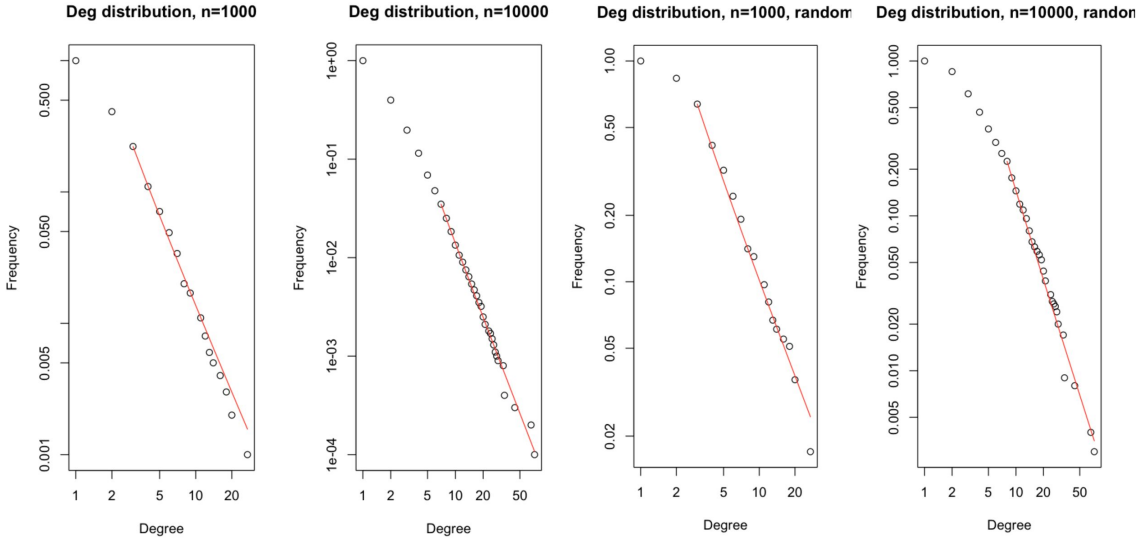


m=5

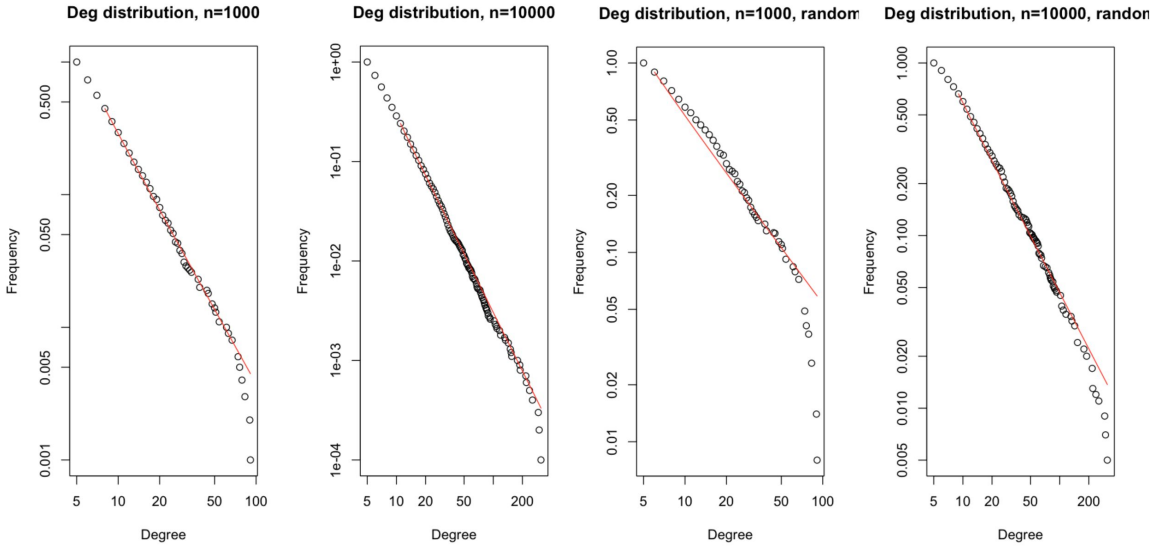
	n = 1000, modularity	n = 10000, modularity
m = 1	0.933	0.978
m = 2	0.527	0.532
m = 5	0.278	0.279

The modularity of the larger network is slightly larger than the modularity of the smaller network for m = 1, m = 2, and m = 5. Another observation is that modularity is decreasing when m is increasing. Notice that the modularity for m = 1 is the largest among three measurements. m is the number of edges each new vertex creates and modularity measures the strength of a network to divide into modules. When m = 1, the new nodes in network has least connections to the old nodes, and thus the network has the strongest strength to be divided into modules. Hence, m = 1 has the largest modularity. Degree distribution plots are shown as below.

m=2

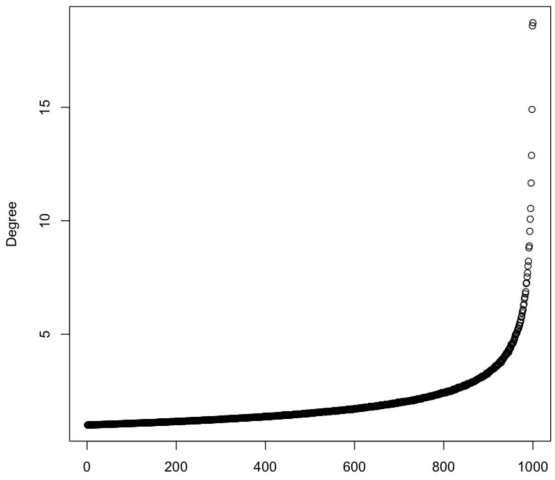


m=5

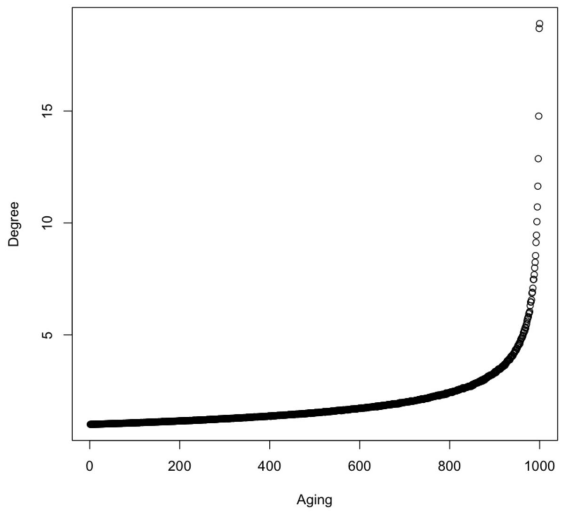


	n = 1000, slope	n = 10000, slope	n = 1000, random, slope	n = 10000, random, slope
m = 2	-3.116	-3.414	-2.390	-2.845
m = 5	-2.850	-2.946	-1.973	-2.078

The slope of 2 leftmost plots is slightly larger and stiffer than the slope of 2 rightmost plots when m = 2 and m = 5. The 2 leftmost plots are the degree distribution of preferential attachment model and the 2 rightmost plots are the degree distribution of uniform picked node at random. Preferential attachment model prefers to link a newly added node to a node that already has high degree. So degrees of older nodes in 2 leftmost plots is greater than that in 2 rightmost plots, which results in heavy tail and relatively stiff slope in 2 leftmost plots. Similarly, degrees of younger nodes in 2 leftmost plots is smaller than that in 2 rightmost plots, which leads to a light tail and relatively less stiff slope in 2 rightmost plots.



m=2

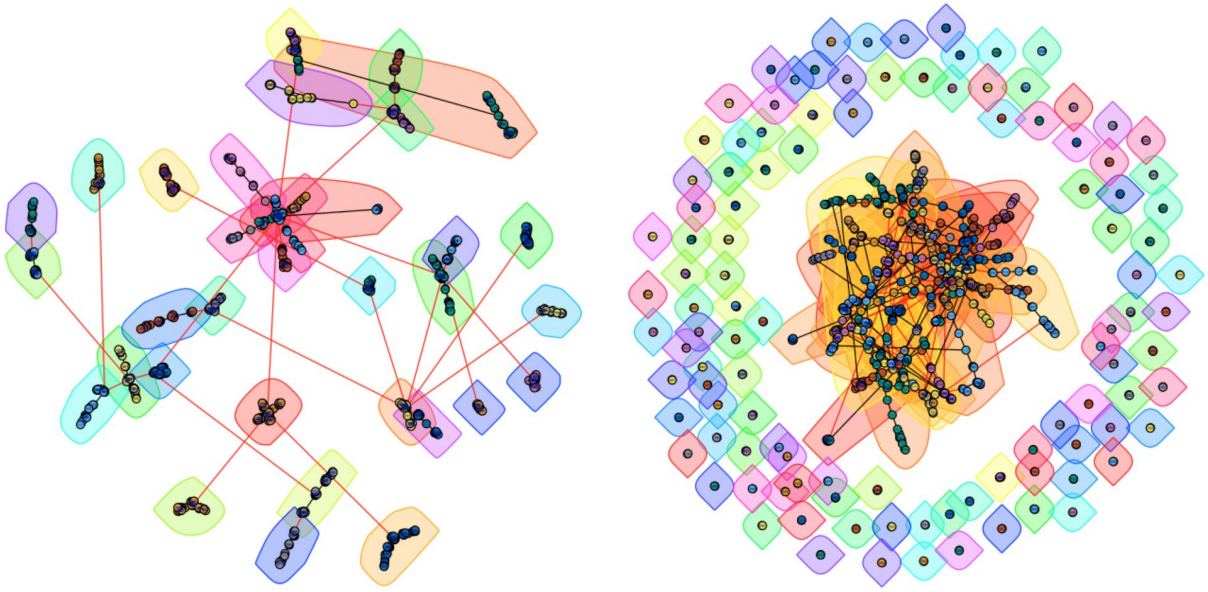


m=5

For both m = 2 and m = 5, nodes with higher degree have stronger ability to grab links added to the network. So older nodes have higher degrees.

(h) We use barabasi.game to generate a preferential attachment network with n = 1000, m = 1 and use sample_degseq and degree functions to create a new network with the same degree sequence.

The modularity of preferential attachment (barabasi) network is 0.932 and the modularity of the network generated by stub-matching procedure is 0.850. Networks and communities are shown in the plots below.



Stub-matching method will randomly match the stubs to connect the nodes of the network by edges after we have generated the nodes of the network along with their degrees. So it is most likely have many connected components. If we create the nodes of the network along with their degrees, then it gives rise to a lot more connected and stable network.

Modularity measures the division strength of the network to be divided into modules. The barabasi network (left plot) has dense connections between nodes within the module and sparse connections between nodes in different modules because it is easy to observe the boundary of different modules. However, the network generated by stub-matching (right plot) has dense connections between nodes in different modules in the center and sparse connections between nodes within the modules at the outer cycle because they only have 1 node. Therefore, right plot (network generated by stub-matching procedure) has smaller modularity than left plot (network generated by the other procedure).

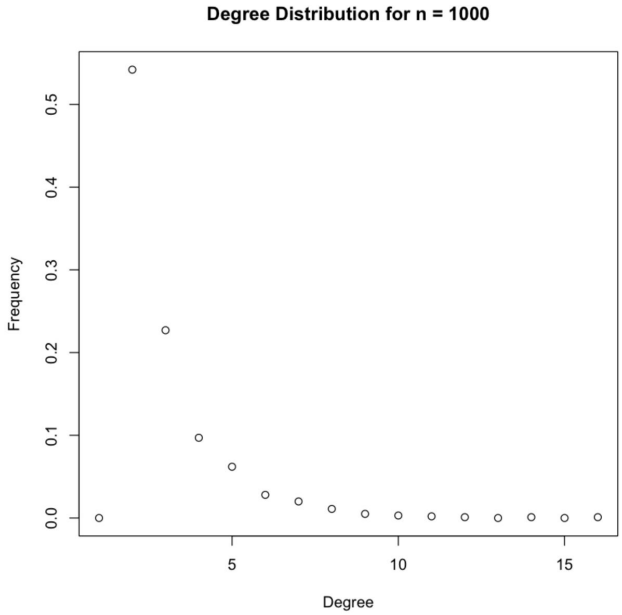
3. Create a modified preferential attachment model that penalizes the age of a node

(a) Each time a new vertex is added, it creates m links to old vertices and the probability that an old vertex is cited depends on its degree (preferential attachment) and age. In particular, the probability that a newly added vertex connects to an old vertex is proportional to:

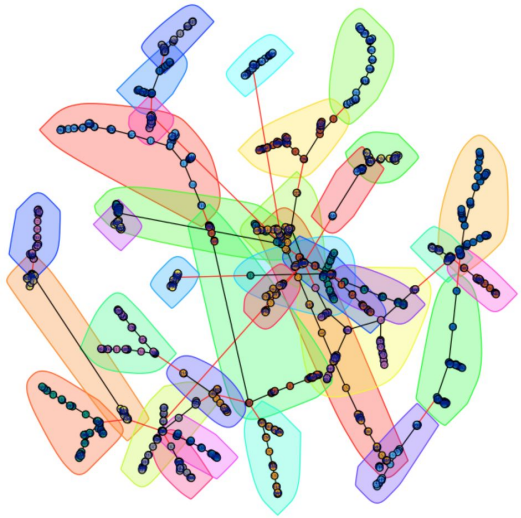
$$P[i] \sim (ck_i^{\alpha} + a)(dl_i^{\beta} + b)$$

where ki is the degree of vertex i in the current time step, and li is the age of vertex i. Produce such an undirected network with 1000 nodes and parameters m = 1, α = 1, β = -1, and a = c = d = 1; b = 0. Plot the degree distribution. What is the power law exponent?

The graph below is the degree distribution of a network with 1000 nodes and penalizing the age of a node. The PA exponent is α = 1 and the aging exponent is β = -1.



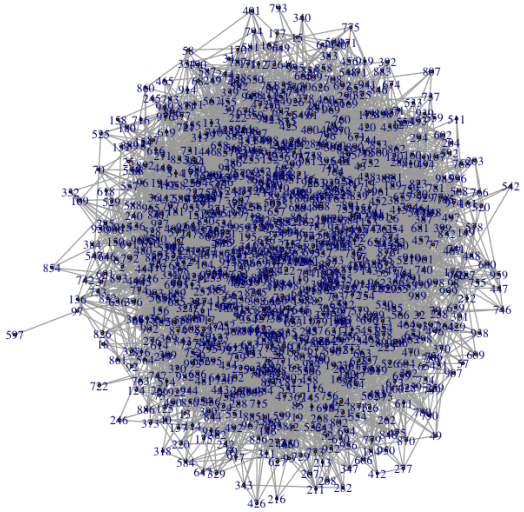
(b) The community structure is shown in the graph below. The modularity is 0.934.



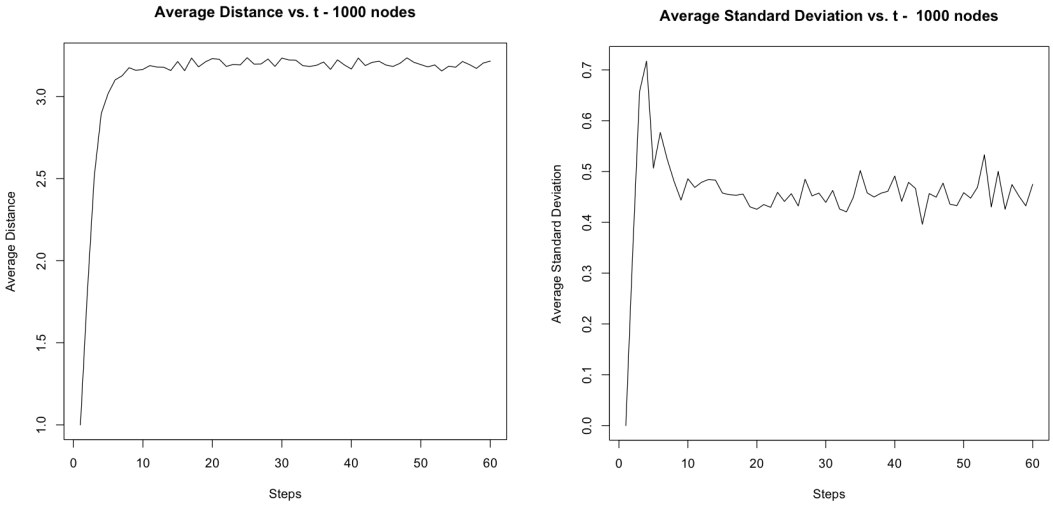
Part 2: Random Walk on Networks

1. Random walk on Erdös-Rényi networks

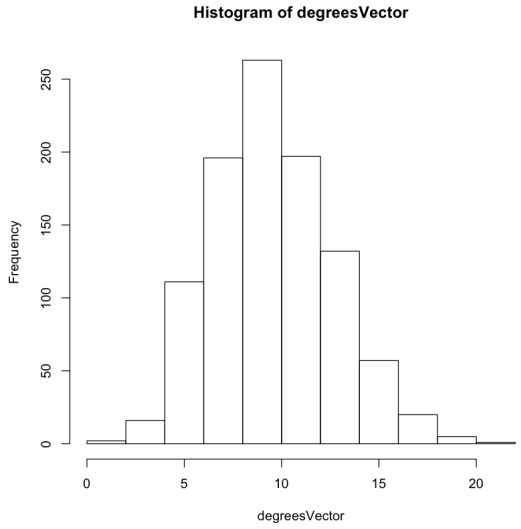
(a) Generated the random graph with 1000 nodes and p for drawing an edge between any pair of nodes equal to 0.01 using the igraph package and erdos.renyi.game. And the graph is connected.



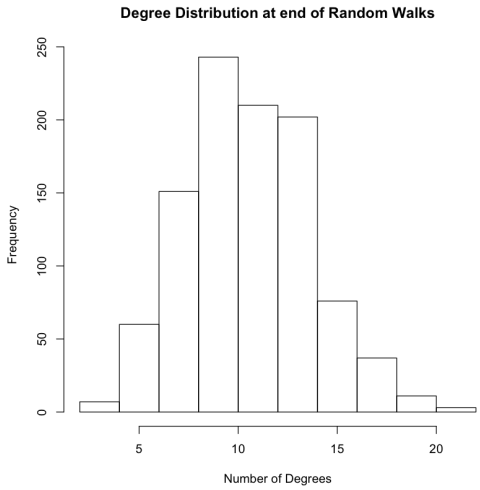
(b) Let a random walker start from a randomly selected node and measure the distance and average distance as well as the standard deviation for each time step t. Here we iterate through all the 1000 nodes and make each node as the starting node. Then we can random walk the graph with the starting node and num of steps to walk and return an end node. The distance is measured by looking at the shortest path between the starting and end node.



(c) The degree distribution of graph is as below:



And the degree distribution of the nodes reached at the end of the random walk is as below:

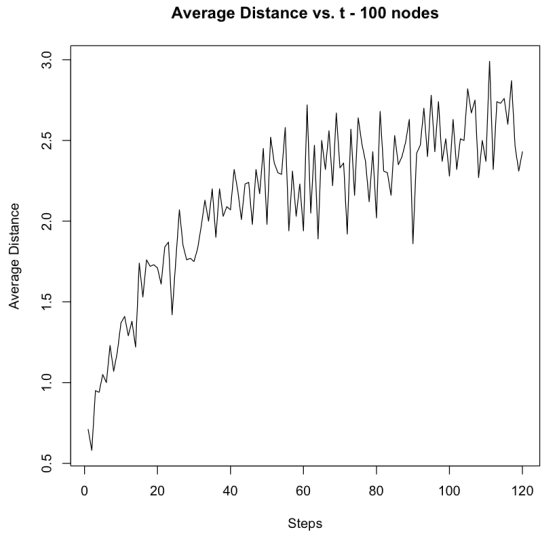


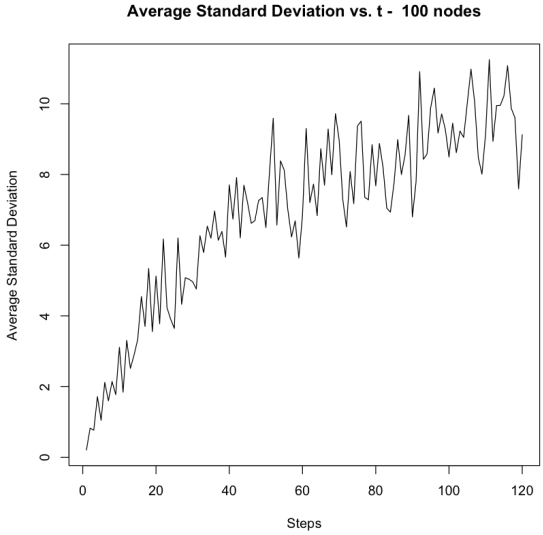
We can tell from the histogram that the 2 degree distribution histograms are very similar. They are both within the range of 0-20 and reach the peak at degree 10-15.

(d) Repeat (b) for undirected random networks with 100 and 10000 nodes.

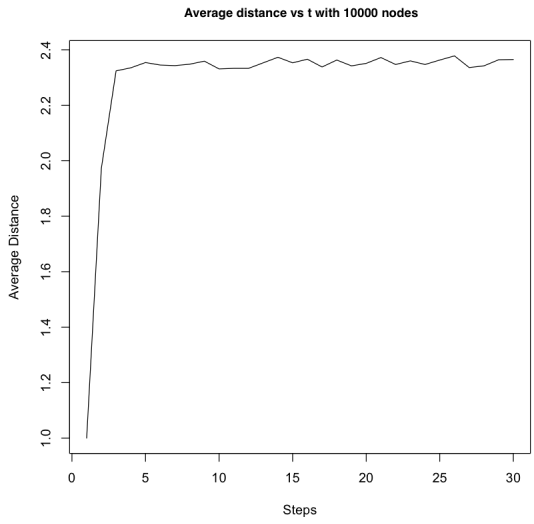
The diameter of graph with 1000, 100, 10000 nodes are 6, 13 and 3. To reduce the running time, we sampled the 10000 nodes by randomly selecting 1000 nodes from them.

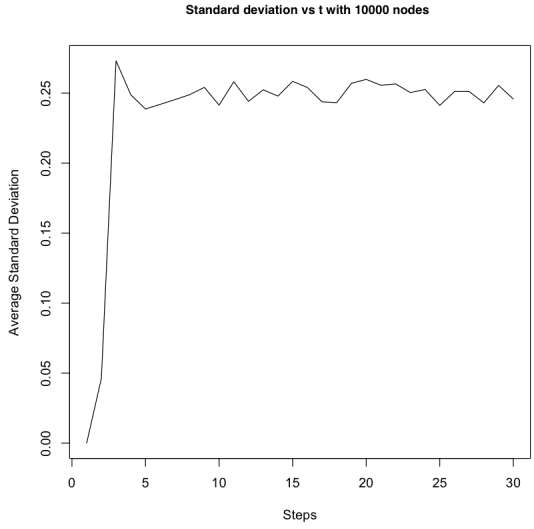
For network with 100 nodes:





For graph with 10000 nodes,





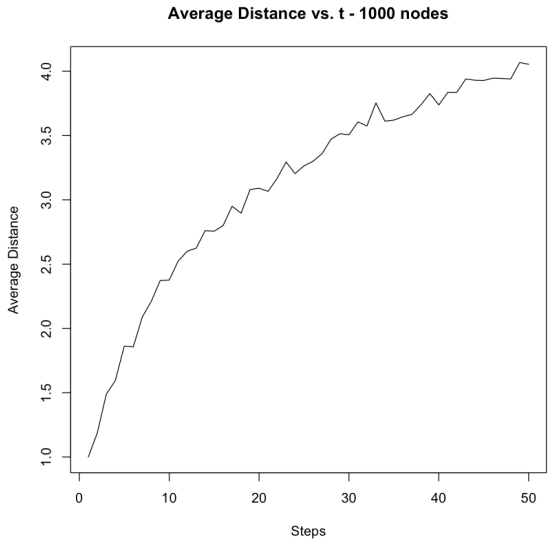
A random network with 100 nodes and p = 0.01 is a very disconnected network. Most walkers are stuck around their starting node. The diameter of the network is 13. On the other hand, a random network with 10000 nodes is denser and has a diameter of 3.

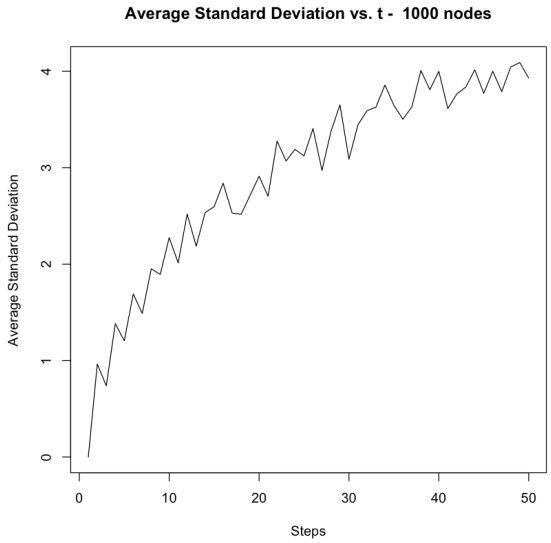
We can tell from the figure that the average distance and standard deviation increases with the steps increase. However, in the case of 10000 nodes, the average distance and standard deviation reached peak at step 3-5 and remain steady after that. The highest distance we can get is around 2.5. And the similar case happened at graph with 100 nodes since the highest average distance is around 10 - 13 which is below the diameter. So we can suspect that the diameter might be the upper bound for the average distance.

2. Random walk on networks with fat-tailed degree distribution

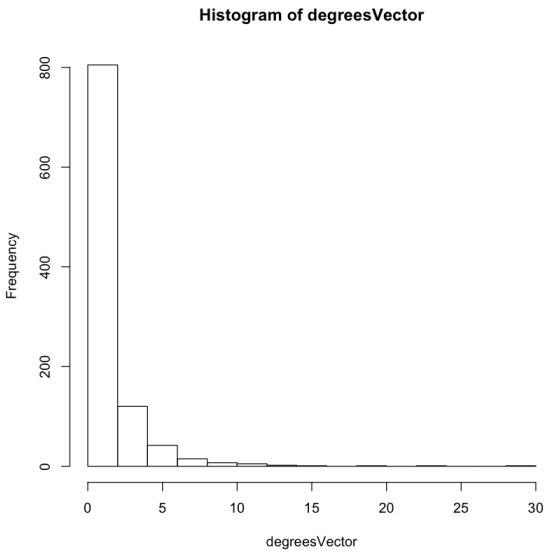
(a) We use barabasi.game from the igraph package to create preferential attachment network with 1000 nodes and each new node attaches to m = 1 old nodes. The network is connected.

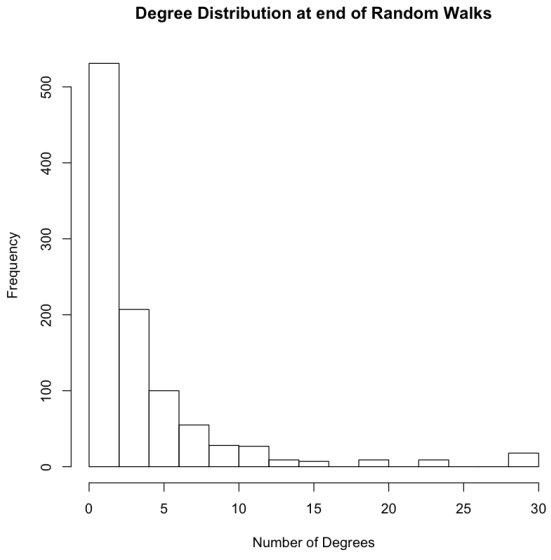
(b)



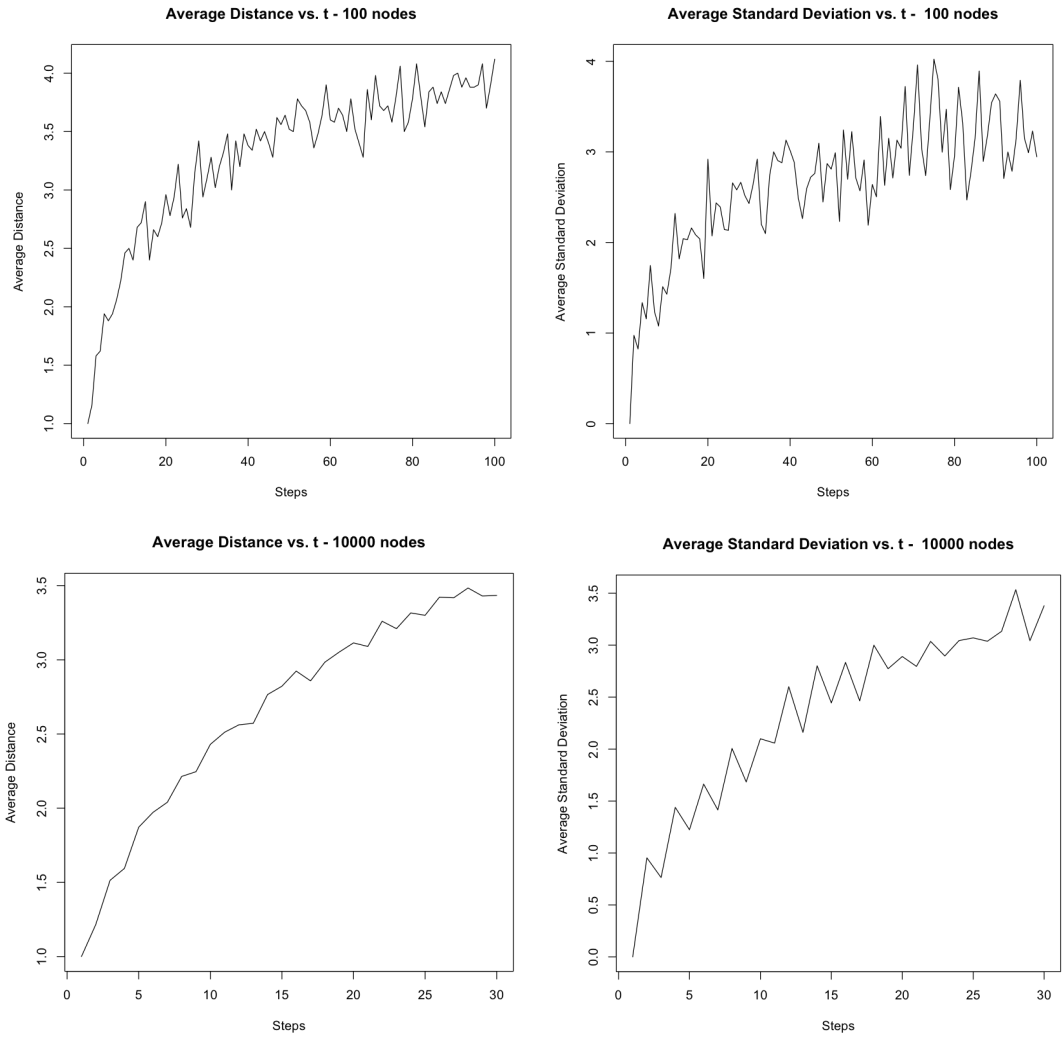


(c) The degree distribution of the nodes from entire graph and the reached nodes at the end of the random walk are very similar. Most of the distribution are in the range of 0-5.



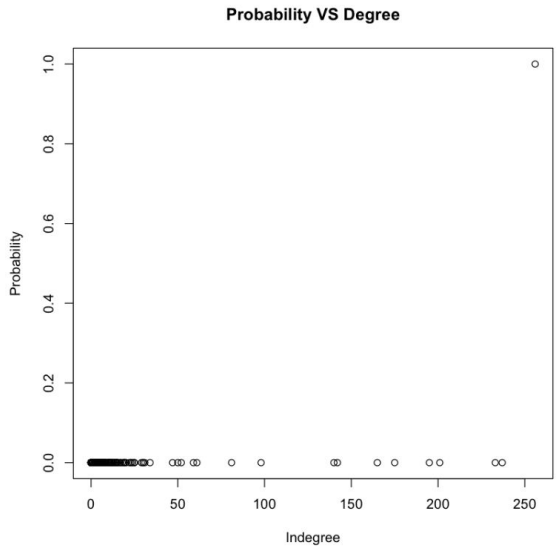


(d) The diameter of 1000, 100, 10000 nodes graph are 20, 11 and 33. We can tell from the figures that the average distance shows a trend to increase with the step size. The reason it doesn't show an obvious upper bound might be because we don't run enough steps to reach the upper bound.

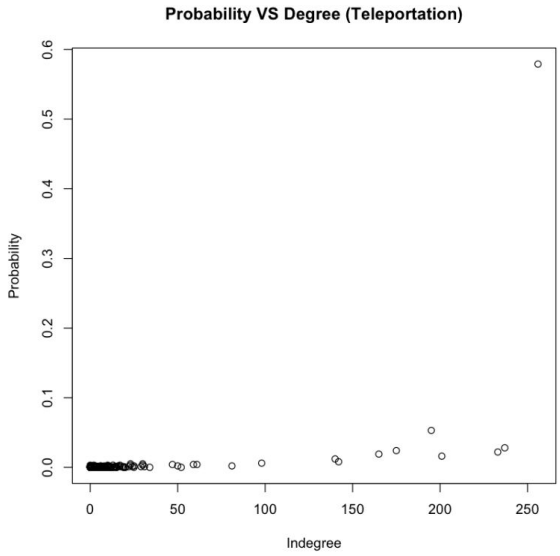


3. PageRank

(a) Using the preferential attachment model, a directed random network is created. After using random walk to simulate PageRank, we measure the probability that the walker visits each node. We can see that all the paths end at the first node of the network, which has much larger degree than others.

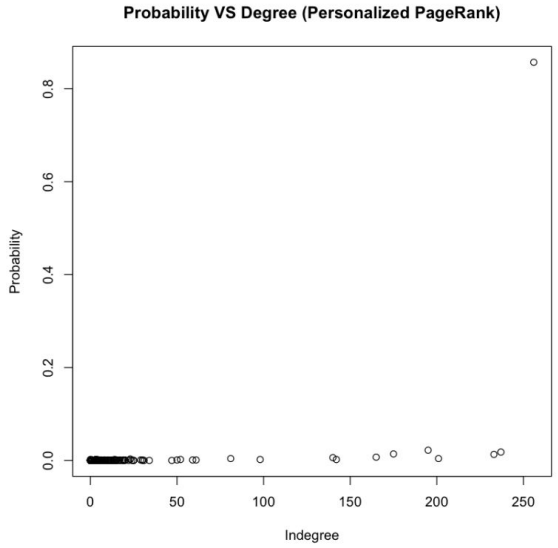


(b) On the same network, random walk with a teleportation probability of 0.15. We can see the probability of paths ending at the first node is still much larger than those of other nodes. As for other nodes, nodes with larger indegree have larger probability.

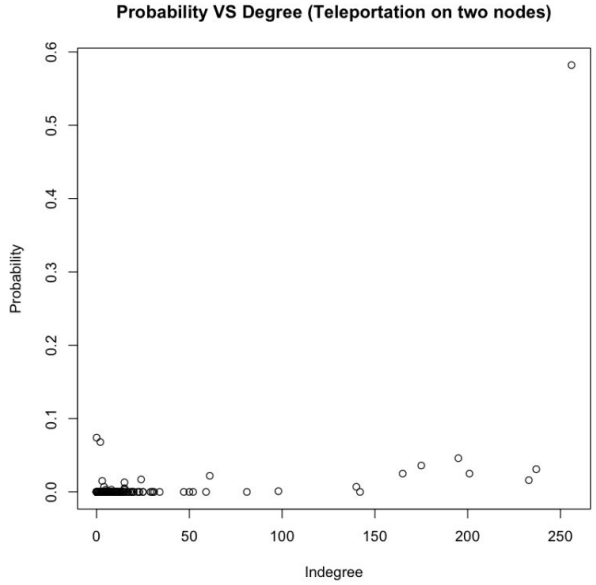


4. Personalized PageRank

(a) Random walk on the same network with teleportation probability of 0.15. Instead of visiting all nodes with same probability when teleportation, let that chance be proportional to the node's PageRank. Compared with the previous result, the probability of a path ending at the first node becomes larger, because in this case, when teleportation, the random walk have a big chance to get to the first node since it has a much larger PageRank than others.



(b) Since only several nodes have a PageRank larger than 0, so we choose two nodes with a medium PageRank which equals to 0. And when teleportation, it has an equal probability to land on one of the two nodes. We can see that in this way the two nodes have a noticeable increase in probability while the probabilities of others have no big changes and the first node (having the largest degree) still has the largest probability.



(c) In the real world, random walk may only teleports to a set of trusted nodes. Taking into account the effect of this self-reinforcement, we can adjust the PageRank equation to the following one, in which *F* is the set of trusted nodes.

$$P_F = (1-d)T + d[1 \ \dots \ 1]^T j_F \text{ with } j_{F_i} = \begin{cases} 1/|F| : i \in F \\ 0 : i \notin F \end{cases}$$