ECE232 Project1
Random Graphs and Random Walks

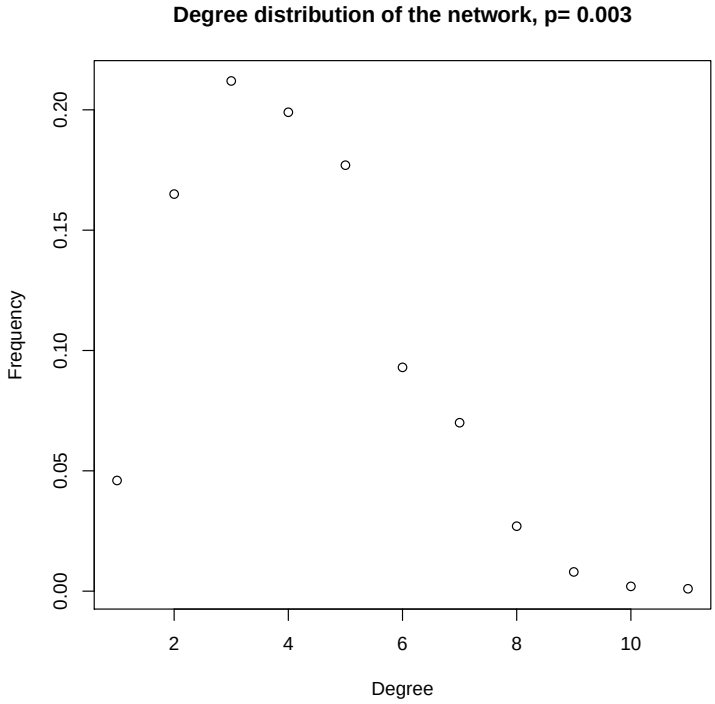
Juo-Wen Fu 805025223
Alex Hung 705035114
Te-Yuan Liu 805027255
Wesly Tzuo 704946416

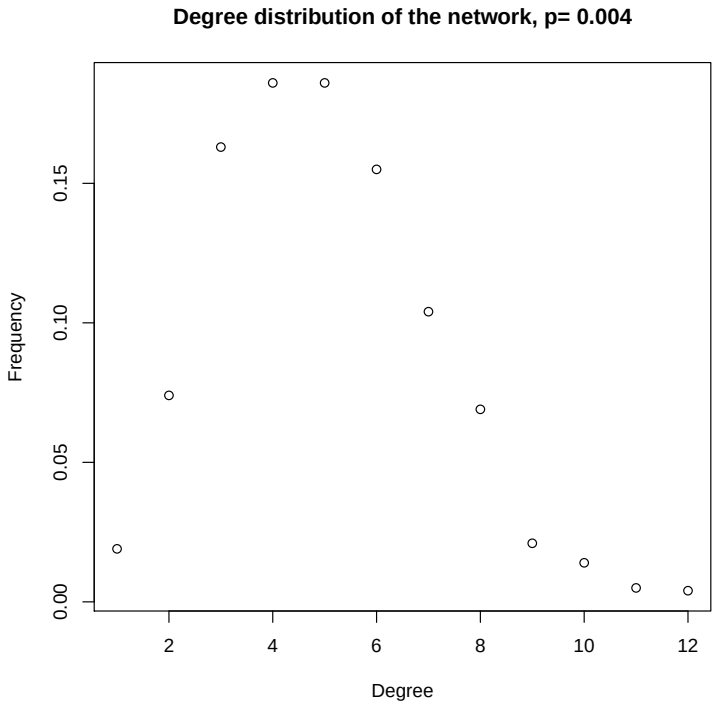
March 19, 2018

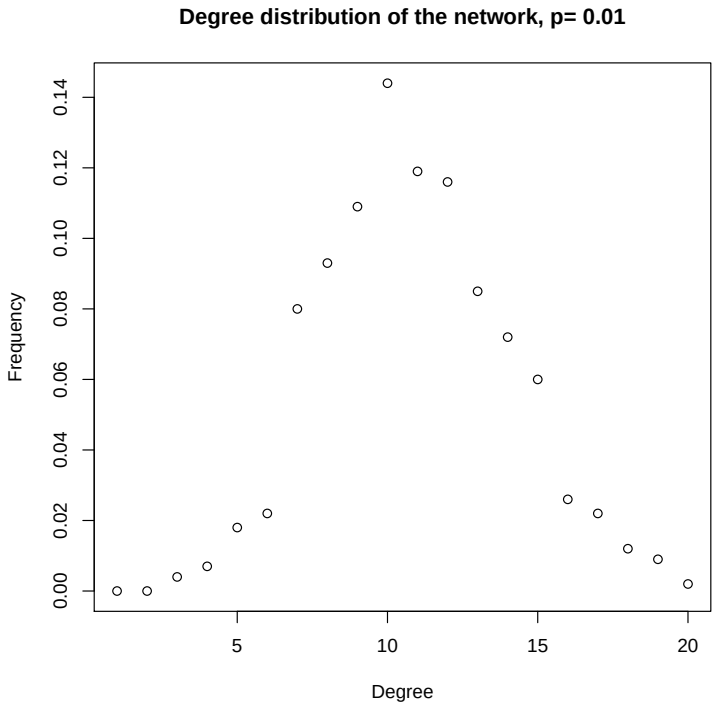
1 Generating Random Networks

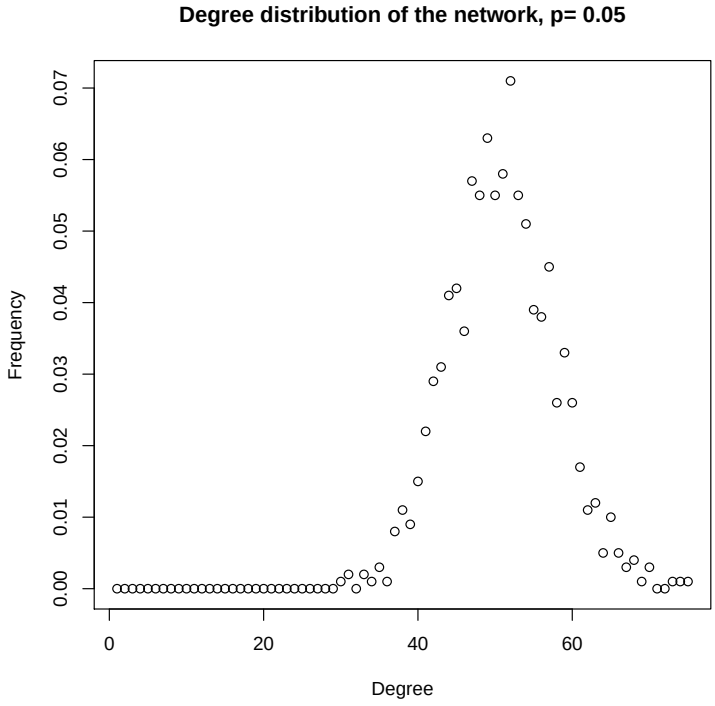
1. Create random networks using Erds-Rnyi (ER) model

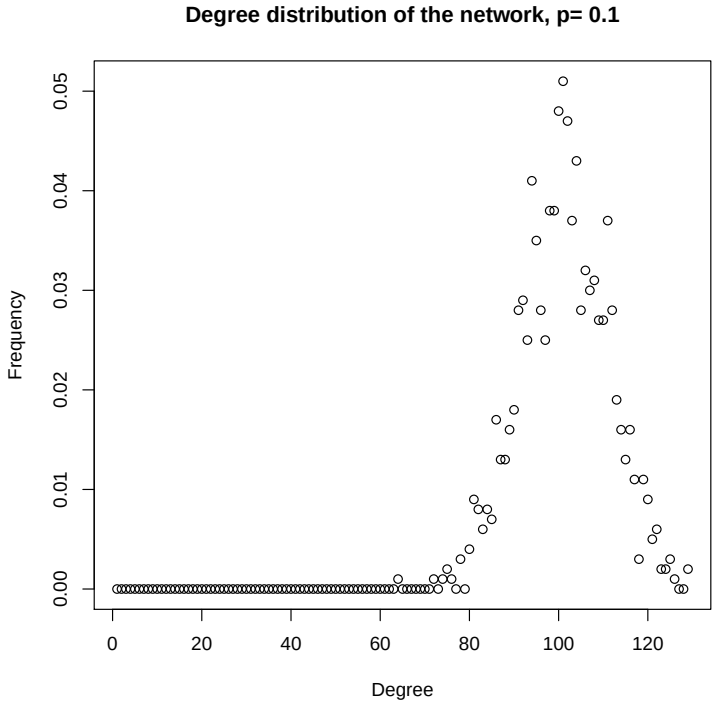
(a) Create an undirected random networks with n = 1000 nodes, and the probability p for drawing an edge between two arbitrary vertices 0.003, 0.004, 0.01, 0.05, and 0.1. Plot the degree distributions. What distribution is observed? Explain why. Also, report the mean and variance of the degree distributions and compare them to the theoretical values.




Degree distribution of the network, p= 0.004
Frequency
0.00
0.05
0.10
0.15
2
4
6
8
10
12
Degree


Degree distribution of the network, p= 0.01
Frequency
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
5
10
15
20
Degree


Degree distribution of the network, p= 0.05
Frequency
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0
20
40
60
Degree


Degree distribution of the network, p= 0.1
Frequency
0.00
0.01
0.02
0.03
0.04
0.05
0
20
40
60
80
100
120
Degree

From the figures shown above, we can observe a *binomial distribution.*
Since with n+1 nodes in total, the probability of observing a node with degree k is:

$$P(deg(v) = k) \ = \ \binom{n}{k} \cdot p^k (1-p)^{n-k}$$

Which is the same as the Probability mass function of *binomial distribution.*

According to *binomial distribution*:

$$mean = n * p$$

$$variance = n * p * (1-p)$$

where n is the number of nodes, p is the probability for drawing an edge between two arbitrary vertices. We can compare our results with the theoretical values.
The table below shows the comparison, both means and variances meet the theoretical values.

p	0.003	0.004	0.01	0.05	0.1
mean	3.06	3.97	9.91	49.87	100.25
var	3.22	4.06	9.32	45.45	93.65
theoretical mean	3	4	10	50	100
theoretical var	2.991	3.98	9.9	47.5	90

(b) For each p and n = 1000, answer the following questions: Are all random realizations of the ER network connected? Numerically estimate the probability that a generated network is connected. For one instance of the networks with that p, find the giant connected component (GCC) if not connected. What is the diameter of the GCC?

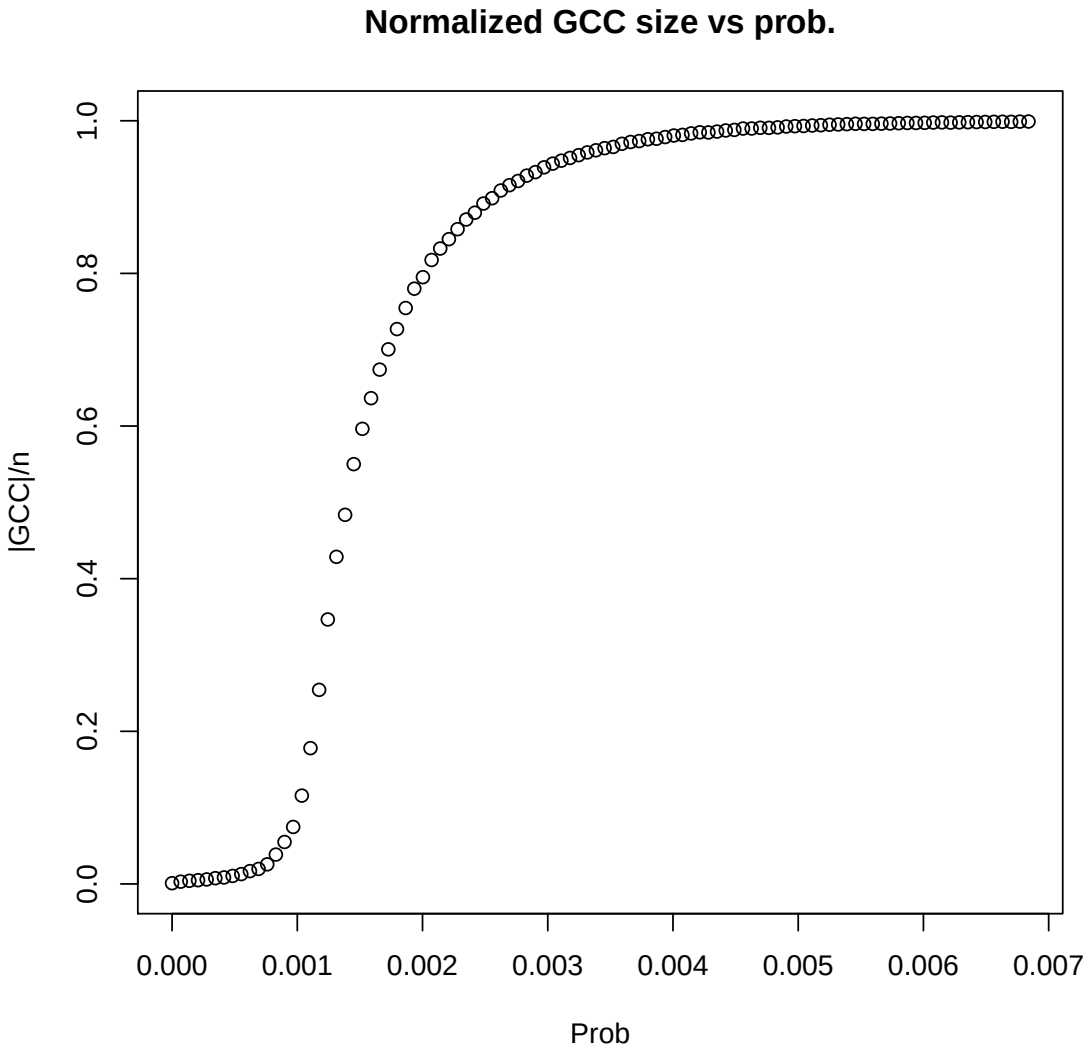
No, not all random realizations of the ER network connected.
Here we generate the network 100 times to estimate the probability that a generated network is connected.
The table below show the empirical probability that a generated network is connected, the diameter and the number of vertex of giant connected component (GCC) if not connected.

	0.003	0.004	0.01	0.05	0.1
Connected prob.	0	0	0.95	1	1
GCC Diameter	15	11	6	-	-
GCC # of vertex	942	985	999	-	-

The $\frac{\ln n}{n}$ is a sharp threshold for the connectedness of Erds-Rnyi (ER) model. For $n = 1000$, $threshold = 0.007$ and the table above verifies this property.

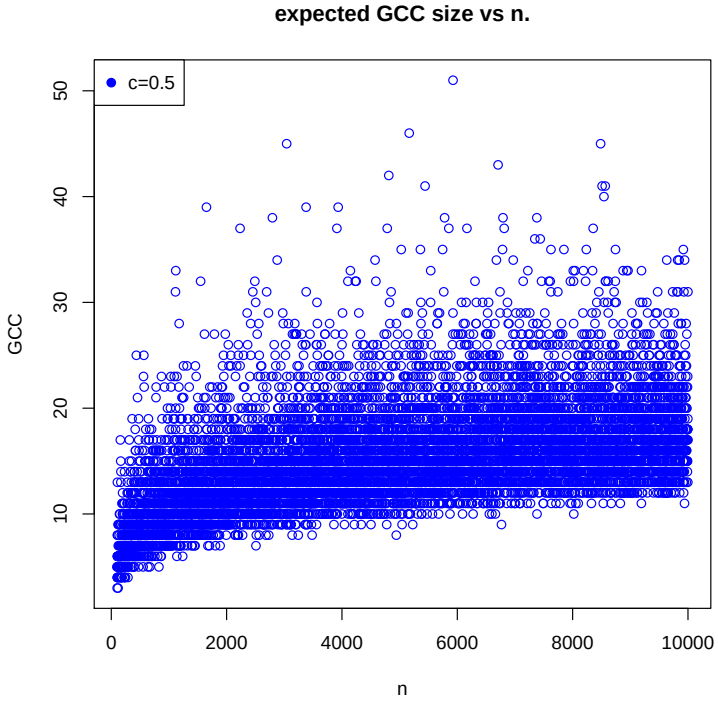
(c)It turns out that the normalized GCC size (i.e., the size of the GCC as a fraction of the total network size) is a highly nonlinear function of p, with interesting properties occurring for values where $p = O(\frac{\ln n}{n})$. For n = 1000, sweep over values of p in this region and create 100 random networks for each p. Then scatter plot the normalized GCC sizes vs p. Empirically estimate the value of p where a giant connected component starts to emerge (define your criterion of emergence)? Do they match with theoretical values mentioned or derived in lectures?

As you can see from the figure below, the GCC sizes are close to 0 until $p = 0.001$. Which meets the values mentioned in lectures that the GCC size will start growing when $n * p = 1$ (Here n = 1000).
The size of GCC starts to emerge roughly when p = 0.003. Here we define "emergence" when the change of normalized GCC size is less than 0.01 while we increase p.

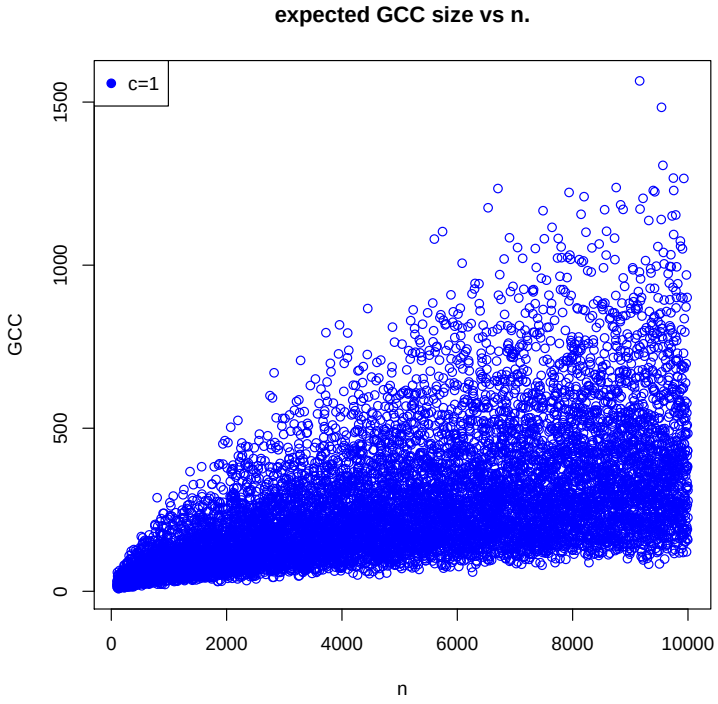


(d) Define the average degree of nodes $c = n * p = 0.5$. Sweep over number of nodes, n, ranging from 100 to 10000. Plot the expected size of the GCC of ER networks with n nodes and edge-formation probabilities $p = c/n$, as a function of n. What trend is observed?

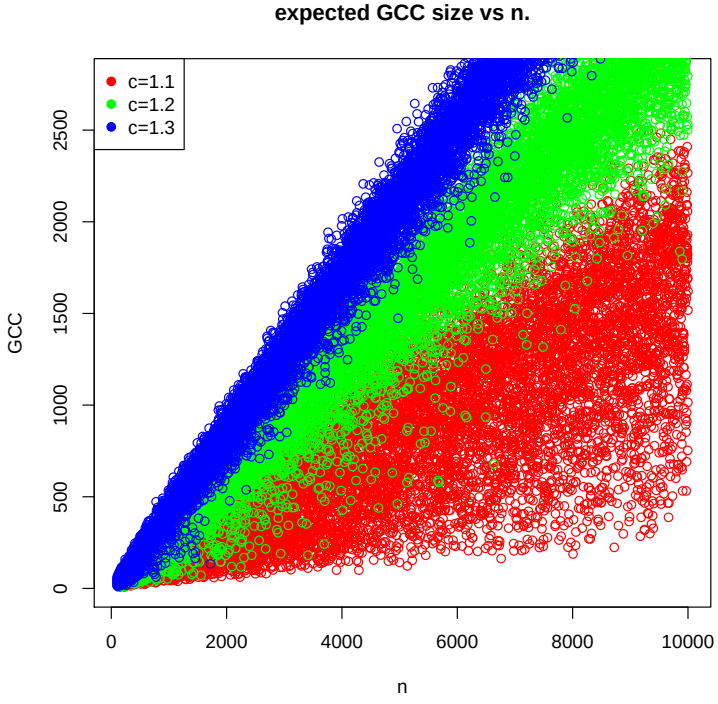
As we mentioned in *Question 1-1 (b)*, the GCC size will be close to 0 if $n * p < 1$.
You can see from the figure below that the GCC size is very small, but still increase while we increase the total number of nodes.



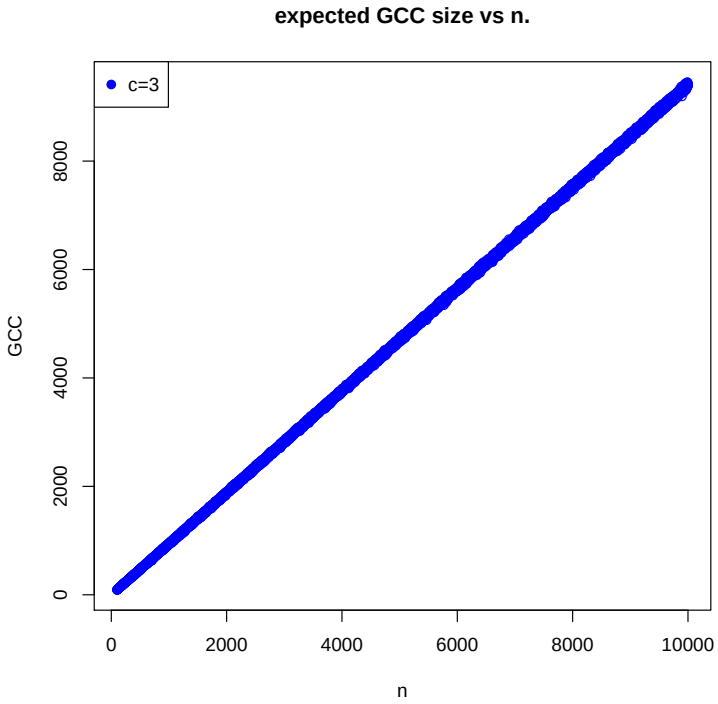
Repeat the same for $c = 1$.
As we increase the $c = n * p$ to 1, which achieve the threshold we discussed in *Question 1-1 (b)*. The size of GCC has significant growth compare to $c = 0.5$.



Repeat the same for values of $c = 1.1, 1.2, 1.3$**, and show the results for these three values in a single plot.** As you can see from the figure, the GCC size start to emerge with larger c, we can see that as we increase the value of c, the GCC size will become closer to the diagonal from bottom left to top right, which represent the fully connected graph ($GCC\ size = n$).



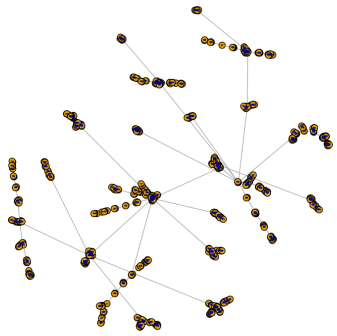
Since we observed in *Question 1-1 (c)* that the GCC size will start to emerge when $p = 0.003$ (where $n = 1000$), here we can set $c = n * p = 1000 * 0.003 = 3$ to verify this trend.



As shown from the figure above, the GCC size becomes a diagonal exactly.

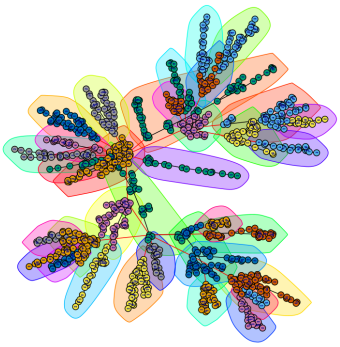
2. Create networks using preferential attachment model

(a) Create an undirected random networks with $n = 1000$ nodes, with preferential attachment model, where each new node attaches to $m = 1$ old nodes. Is such a network always connected?



The network generated is shown in the above graph. Networks generated using preferential attachment model are always connected, because every incoming node has to connect to the existing network with m edges.

(b) Use fast greedy method to find the community structure. Measure modularity.

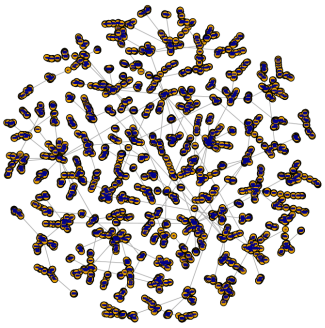


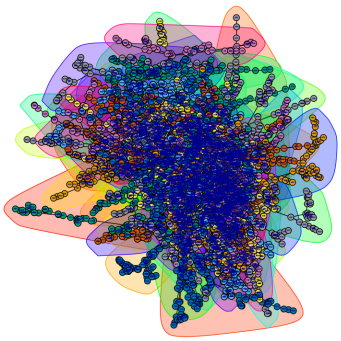
The community structure is shown in the above graph. Because the number of nodes are not that huge, we can see some clear boundaries of the communities. The formula of the modularity of a network is given below.

$$Q = \frac{1}{2m}\sum_{i,j}((A_{ij} - \frac{k_i k_j}{2m})\delta(c_i, c_j)) \tag{1}$$

In the above equation, m is the number of edges, k_i the degree of node i, k_j the degree of node j, c_i the community to which node i belongs, c_j the community to which node j belongs. Note that $\delta(c_i, c_j)$ is one if node i and node j are in the same community, and it is zero otherwise.
The modularity of the generated network is 0.931698. This high modularity implies that most of the nodes in the same community have edges connecting to each other and have few edges connecting to other communities.

(c) Try to generate a larger network with 10000 nodes using the same model. Compute modularity. How is it compared to the smaller networks modularity?

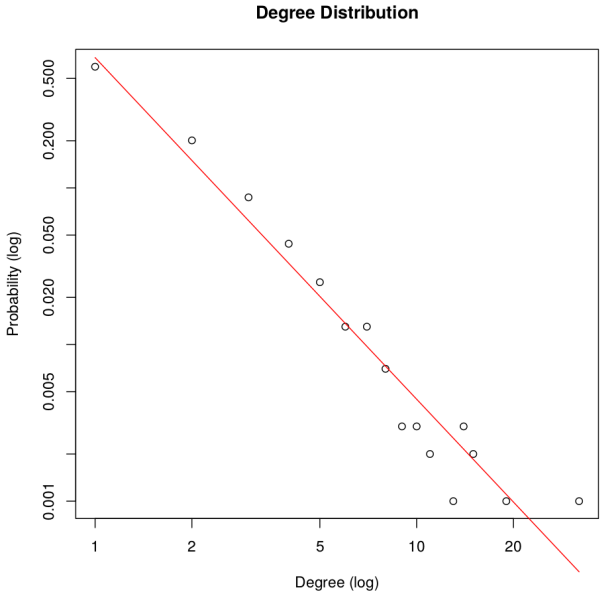




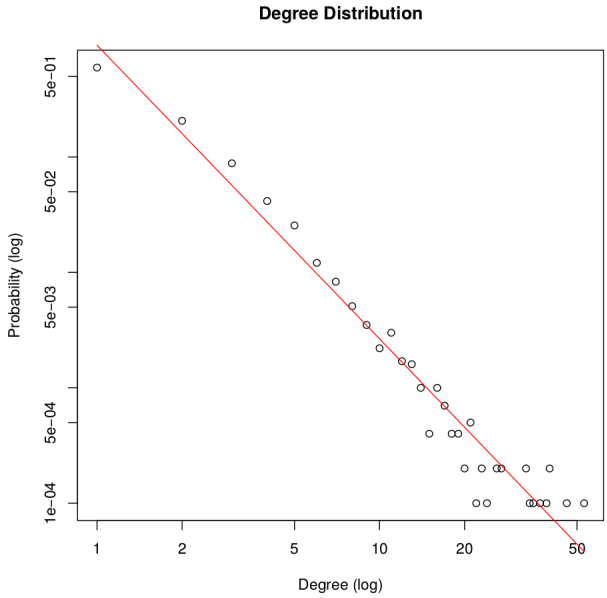
The network plot and the community structure plot are shown above. The modularity of the network is 0.9783221, which is greater than the smaller network's modularity. This indicates that compared to the smaller network, more communities are more well-separated and there is a stronger connection between nodes within each community.

(d) Plot the degree distribution in a log-log scale for both $n = 1000$, 10000, then estimate the slope of the plot.

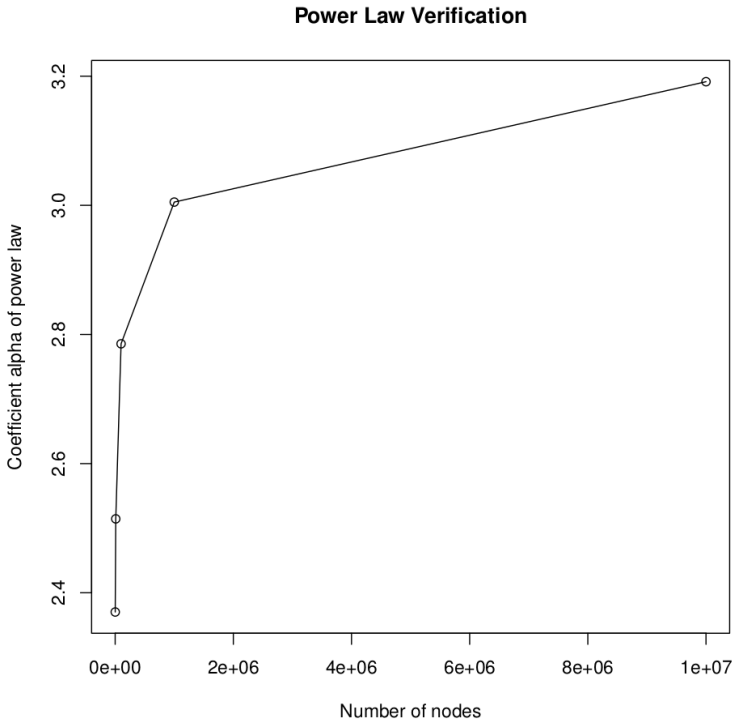
For $n = 1000$:



The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 2.181 and the root mean square error of the fitting line is 0.942.
For $n = 10000$:



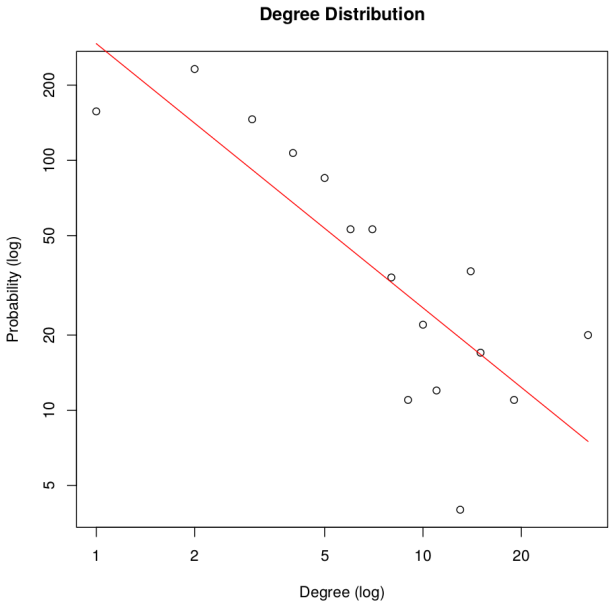
The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 2.544 and the root mean square error of the fitting line is 0.956.



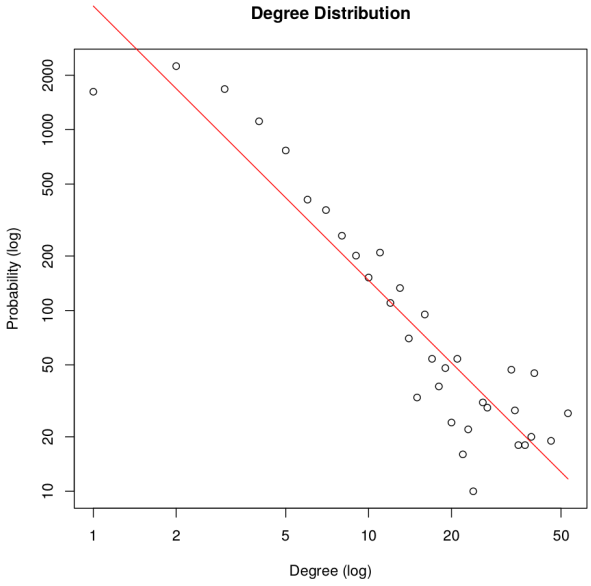
In the lectures, we have derived that the degree distribution frequency of degree k becomes proportional to $\frac{1}{k^3}$ for networks with a lot of nodes. Now, we have conducted an experiment to verify the power law of degree distribution and the result is shown in the above graph. From the plot, we know that as the number of nodes approaches 10^6, the power exponent, or coefficient alpha, is getting closer to our theoretical value, which is three, and then it exceeds three in a very slow manner. This experiment is an empirical proof for the derivation of power law.

(e) You can randomly pick a node i, and then randomly pick a neighbor j of that node. Plot the degree distribution of nodes j that are picked with this process, in the log-log scale. How does this differ from the node degree distribution?

For $n = 1000$:



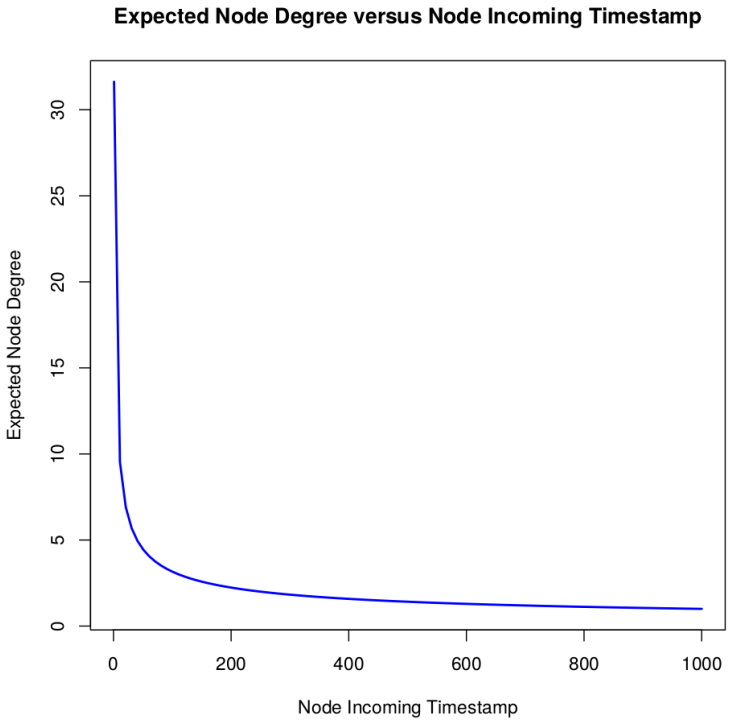
The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 1.057 and the root mean square error of the fitting line is 0.658.
For $n = 10000$:



The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 1.517 and the root mean square error of the fitting line is 0.864.
For a network with n nodes, we conduct the selection n times and then plot the corresponding

degree distribution. Compared to the previous case, this procedure seems to follow the power law but come with a smaller absolute power exponent value. Because of the random selection and random walk, a node can be selected multiple times so it is difficult to explore every node within the network. Furthermore, nodes with larger degrees are more likely to be picked at last while nodes with smaller degrees are hardly selected. Therefore, the trend of the degree frequency decreases with increasing degree is not that steep, and this is verified by the smaller absolute power exponent value.

(f) Estimate the expected degree of a node that is added at time step i for $1 \leq i \leq 1000$. Show the relationship between the age of nodes and their expected degree through an appropriate plot.

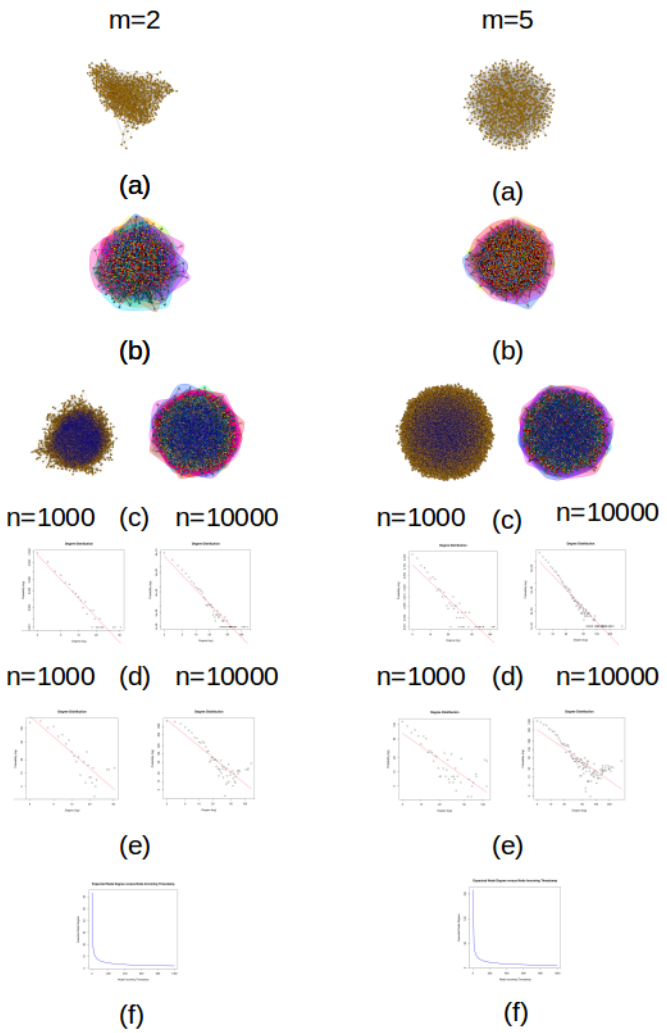


The plot of the relationship between the age of nodes and their expected degree is shown above. We use the equation derived in the lectures to arrive at this plot. The equation is given below.

$$k(i,t) = m(\frac{t}{i})^{\frac{1}{2}} \tag{2}$$

In the equation, $k(i,t)$ is the average degree of the node added at the ith step at timestamp t, and m is the number of edges every incoming node creates to attach to the existing network.
From the above graph, we know that the vertex existing from the very beginning benefits the most from the growth of the network as it has the highest degree and connects to more vertices than any other nodes.

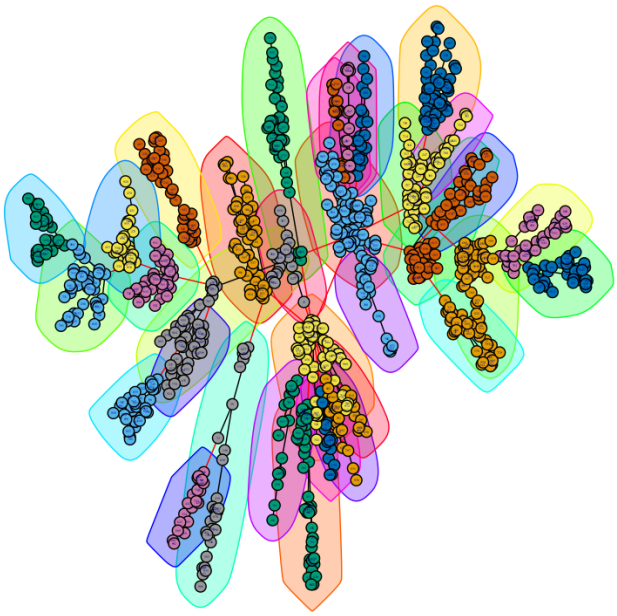
(g) Repeat the previous parts for $m = 2$, and $m = 5$. Why was modularity for $m = 1$ high?

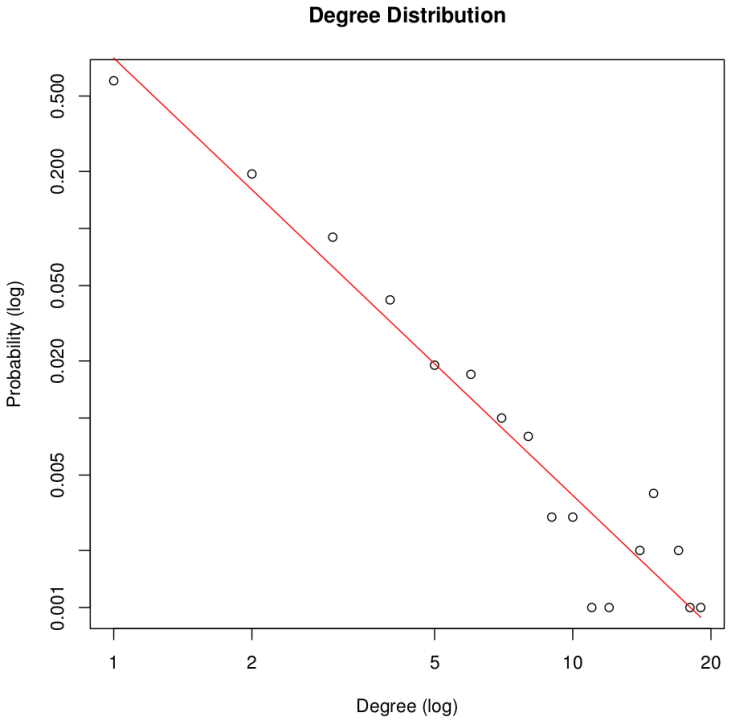


(a) The figures are shown above.
(b) For $m = 2$: The modularity of the network is 0.5238933.
For $m = 5$: The modularity of the network is 0.2723007.
(c) For $m = 2$: The modularity of the network is 0.5296102.
For $m = 5$: The modularity of the network is 0.2734247.
(d) For $m = 2$ and $n = 1000$: The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 2.236 and the root mean square error of the fitting line is 0.919.
For $m = 2$ and $n = 10000$: The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 2.349 and the root mean square error of the fitting line is 0.916.
For $m = 5$ and $n = 1000$: The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 1.886 and the root mean square error of the fitting line is 0.828.

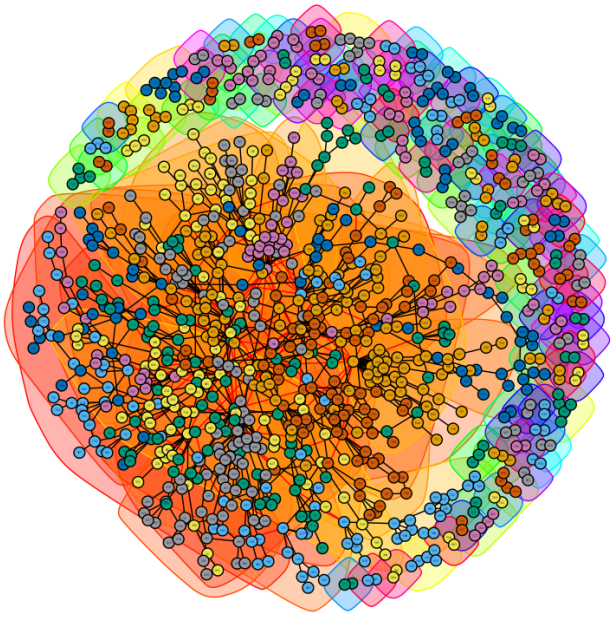
For $m = 5$ and $n = 10000$: The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 2.16 and the root mean square error of the fitting line is 0.895.
(e) For $m = 2$ and $n = 1000$: The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 1.172 and the root mean square error of the fitting line is 0.726.
For $m = 2$ and $n = 10000$: The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 1.36 and the root mean square error of the fitting line is 0.742.
For $m = 5$ and $n = 1000$: The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 0.914 and the root mean square error of the fitting line is 0.603.
For $m = 5$ and $n = 10000$: The degree distribution plot in log-log scale is shown above. The absolute value of the estimated slope is 1.151 and the root mean square error of the fitting line is 0.656.
(f) The plots of the relationship between the age of nodes and their expected degree are shown above.
The modularity for m equals one is higher than the modularity for m equals two or five because with multiple chances to connect to the existing nodes, incoming nodes can decrease the modularity by connecting to a single node within multiple communities. Thus, a bigger m is likely to contribute to a less modulated network.

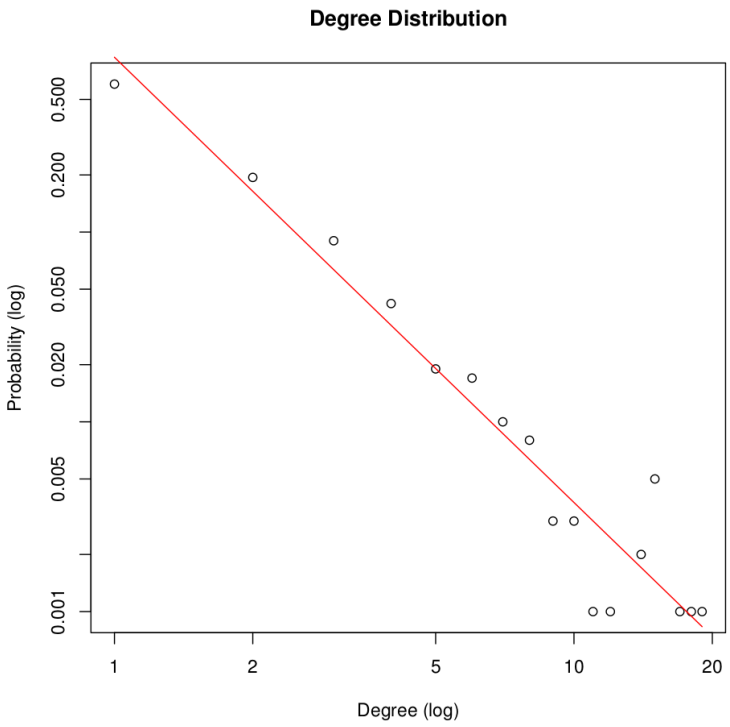
(h) Again, generate a preferential attachment network with $n = 1000$, $m = 1$. Take its degree sequence and create a new net- work with the same degree sequence, through stub-matching procedure. Plot both networks, mark communities on their plots, and measure their modularity. Compare the two procedures for creating random power-law networks.





The plot of the network with marked communities using preferential attachment procedure and the degree distribution plot in log-log scale are shown above. The absolute value of the estimated slope is 2.308 and the root mean square error of the fitting line is 0.932. The modularity of the network is 0.9309019.





The plot of the network with marked communities using stub-matching procedure and the degree distribution plot in log-log scale are shown above. The absolute value of the estimated slope is 2.346 and the root mean square error of the fitting line is 0.932. The modularity of the network is 0.8454358.

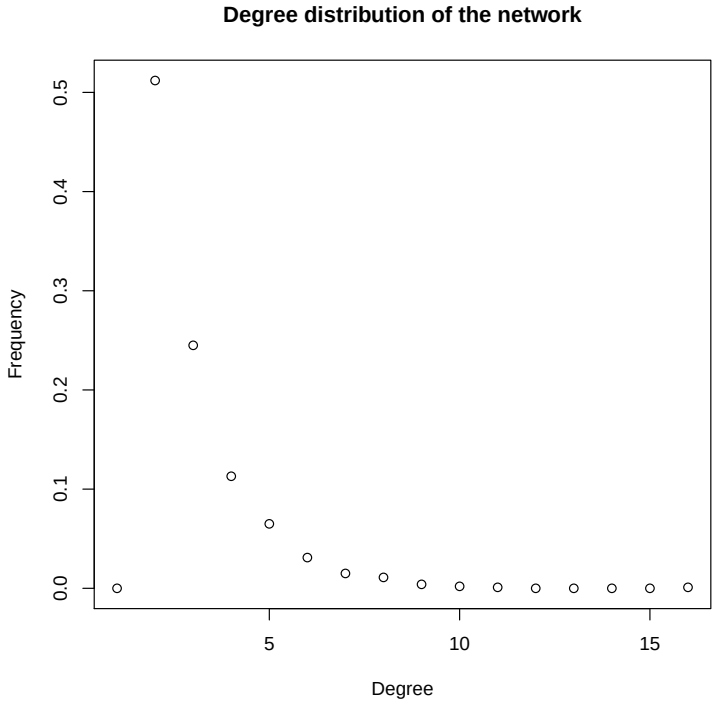
It is obvious that the stub-matching procedure generates a rather more complicated network due to the permutation process. Through the permutation process, the entire network is broken into small parts so it is no longer completely connected. This result may lead to a higher modularity. However, the permutation process is likely to connect high degree nodes together and then generate a large community, which is not preferred when it comes to modularity. Therefore, a lower modularity is achieved by the the network generated using stub-matching procedure. Comparing the degree distribution plots, we know that their power exponents are only slightly different. In overall, we can conclude that the stub-matching procedure does not lead to a huge change in the degree distribution and it is likely to decrease the modularity of the network.

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:

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

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



By using *power.law.fit()* function we can get the power law exponent.
The *power law exponent* = 2.882244.

(b) Use fast greedy method to find the community structure. What is the modularity?

By using *fastgreedy.community()* function we can get the modularity.
The *modularity* = 2.882244.

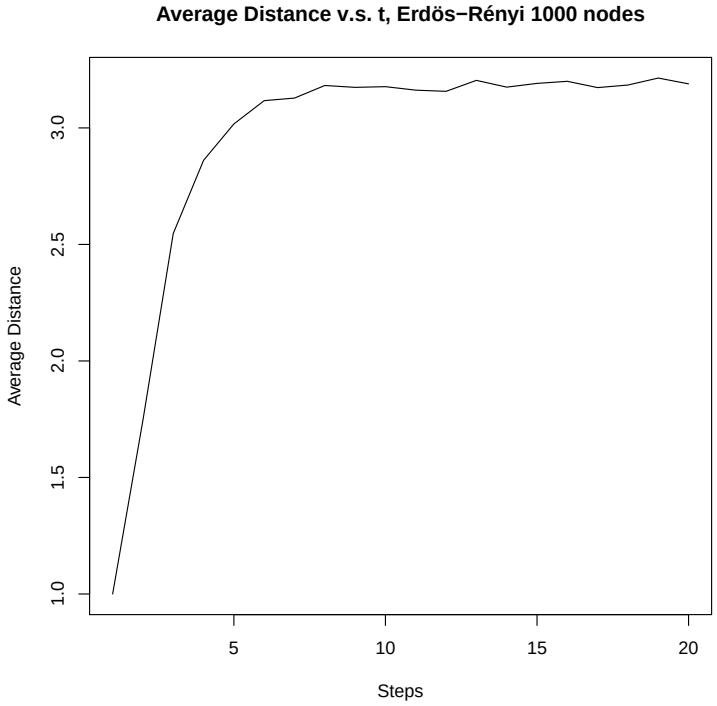
2 Random Walk on Networks

1. Random walk on Erds-Rnyi networks

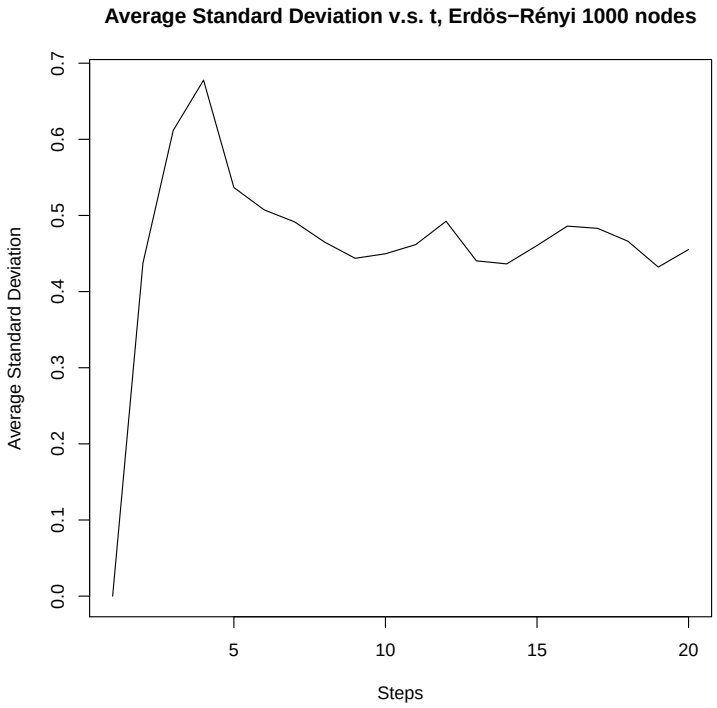
(a) Create an undirected random network with 1000 nodes, and the probability p for drawing an edge between any pair of nodes equal to 0.01.

We used *erdos.renyi.game* in the *igraph* package to create the network.

(b) Let a random walker start from a randomly selected node (no teleportation). We use t to denote the number of steps that the walker has taken. Measure the average distance (defined as the shortest path length) s(t) of the walker from his starting point at step t. Also, measure the standard deviation $\sigma^2(t) =< (s(t) < s(t) >)^2 >$ of this distance. Plot $< s(t) >$ v.s. t and $\sigma^2(t)$ v.s. t. Here, the average is over random choices of the starting nodes.

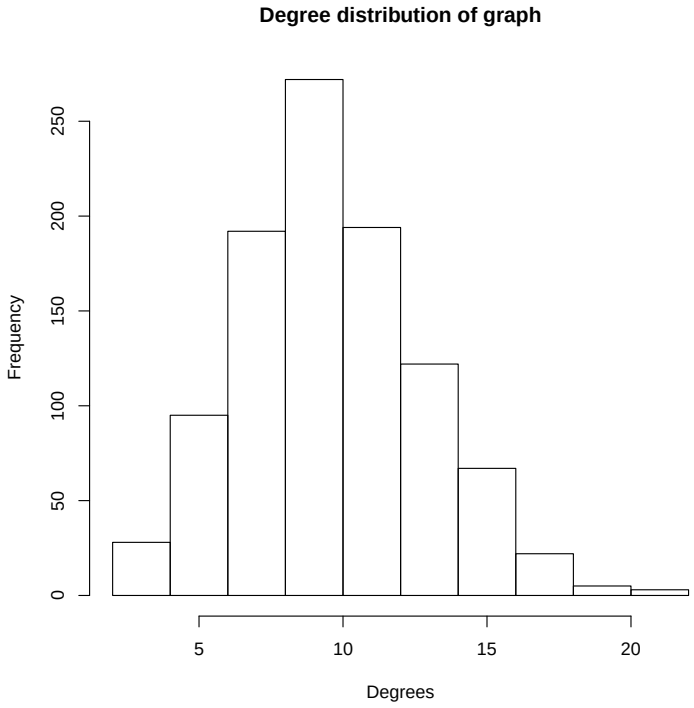


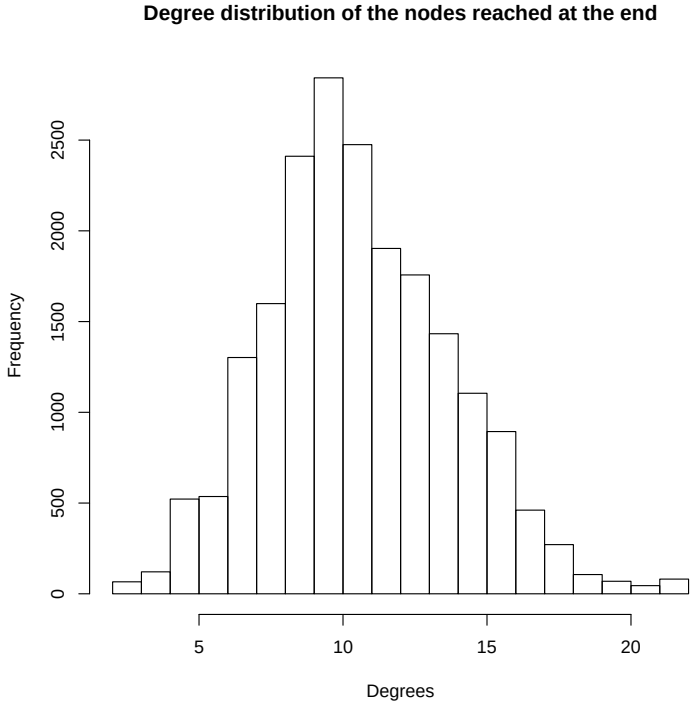
The average distance increases as t increases when t is smaller than 6 and becomes steady after the step is larger than 6. The average distance is around 3 when the step is larger than 6.



The average standard deviation increases as t increases when t is smaller than 4 and slightly decreases when t increases when t is between 4 and 8. Then the average standard deviation becomes steady when step is larger than 8.

(c) Measure the degree distribution of the nodes reached at the end of the random walk. How does it compare to the degree distribution of graph?

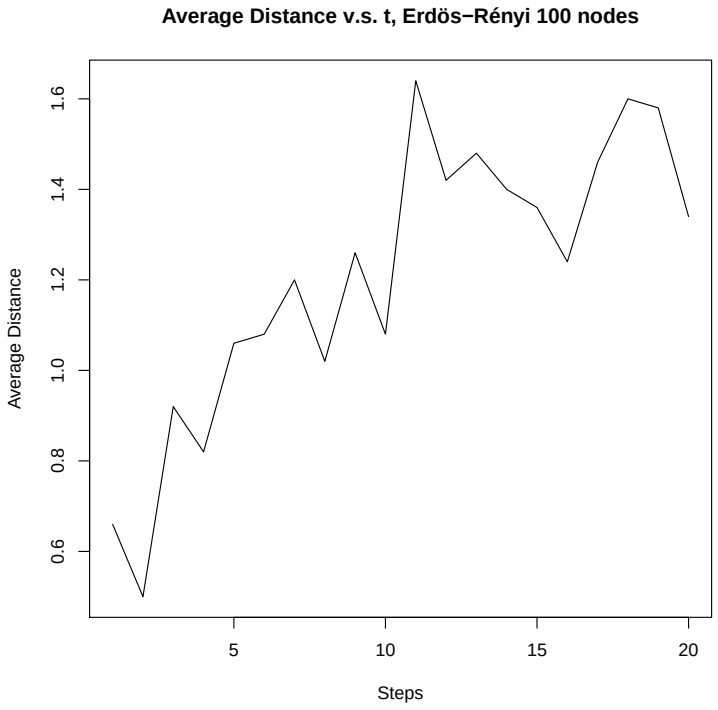




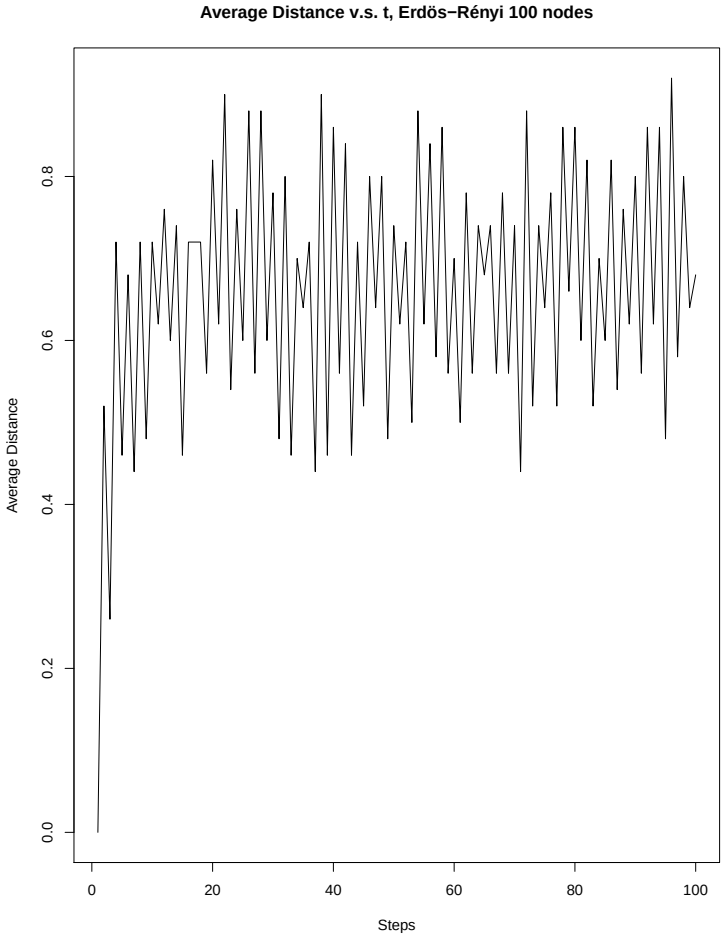
The degree distribution of the graph is similar to the degree distribution of the nodes reached at the end. Most of the number of degrees are between 5 to 15 and few number of degrees are less than 5 or more than 15, which looks like a Gaussian distribution.

(d) Repeat (b) for undirected random networks with 100 and 10000 nodes. Compare the results and explain qualitatively. Does the diameter of the network play a role?

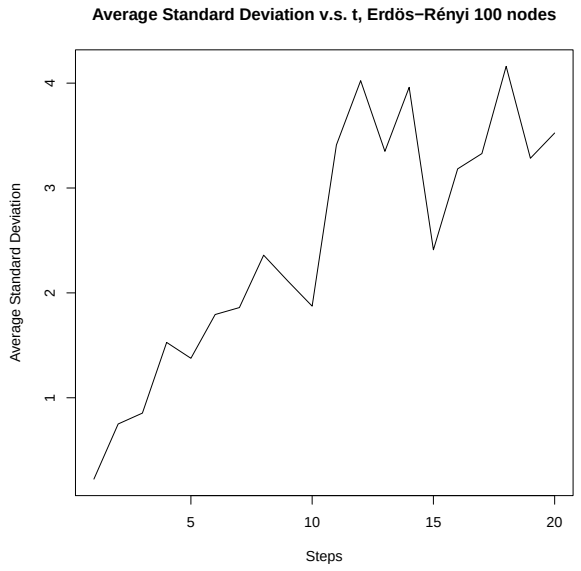
Yes, diameter of the network plays a role.

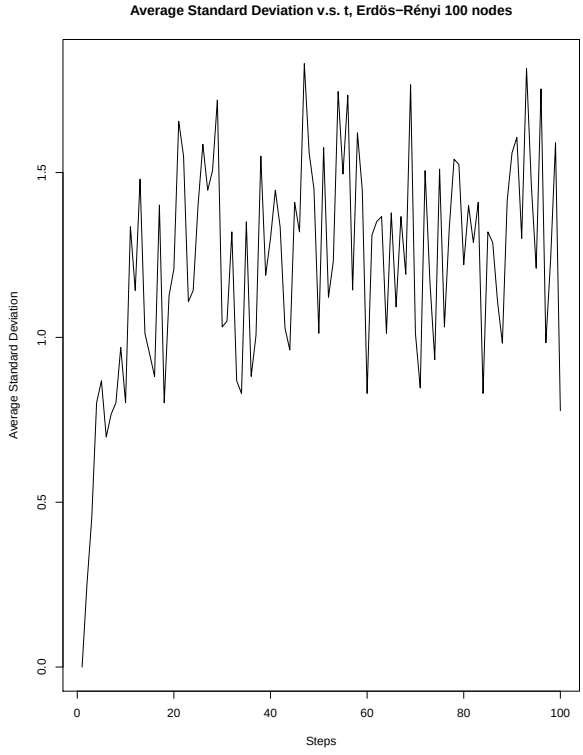


By looking at this figure we are still not sure what the relation between average distance and the step is, so we do more steps to see their relationship.

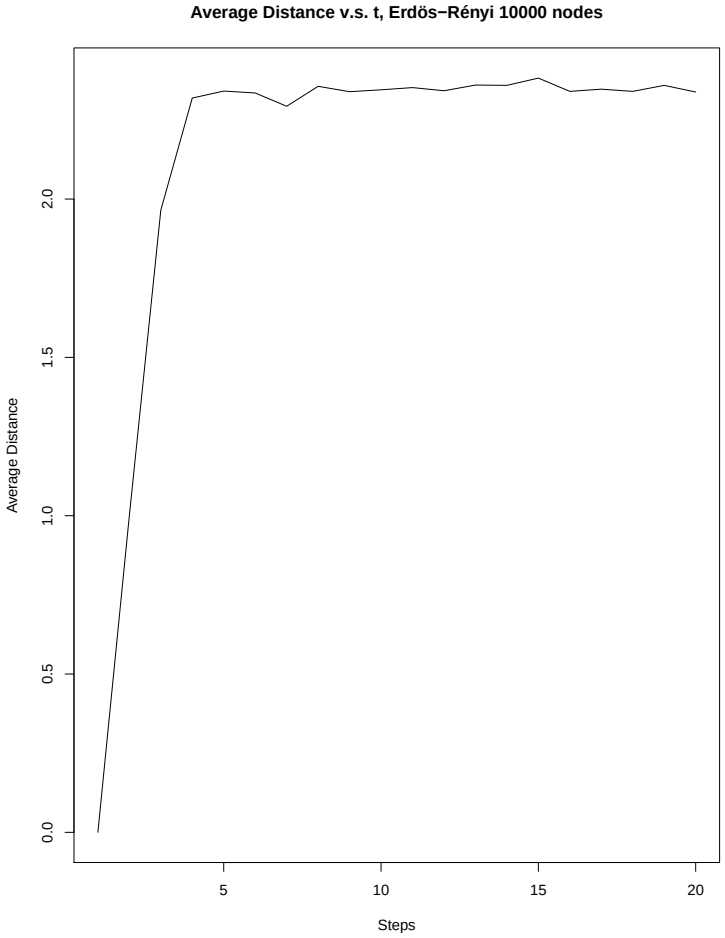


As we can see, the average distance is about 0.4 to 0.8 when there is only 100 nodes. That is, the graph is highly disconnected so when performing random walk, most of the nodes are stuck in itself.



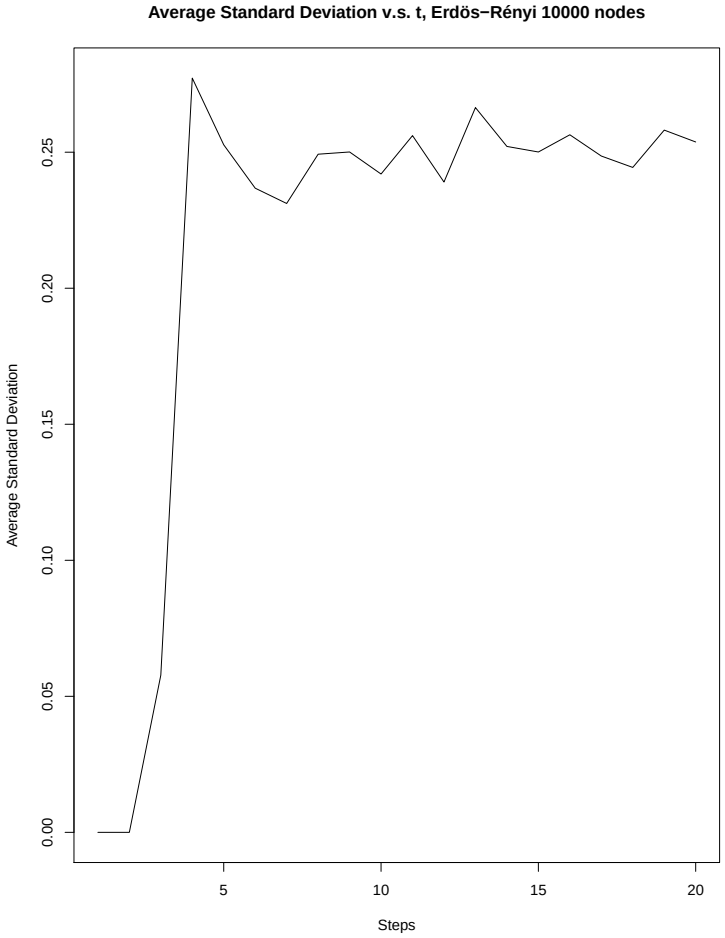


We also do more steps to observe the relationship between t and average standard deviation. While in 1000 nodes, most average standard deviations are lower that 1, in 100 nodes, most average standard deviations are higher than 1.



For 10000 nodes, at each step we sample 1000 nodes to do random walk.

Comparing to 1000 nodes, in most steps 10000 nodes has average distance of 2.5. The trend between the average distance and steps is similar to the trend of 1000 nodes. The average distance increases as t increases when t at a number and becomes steady after that number.



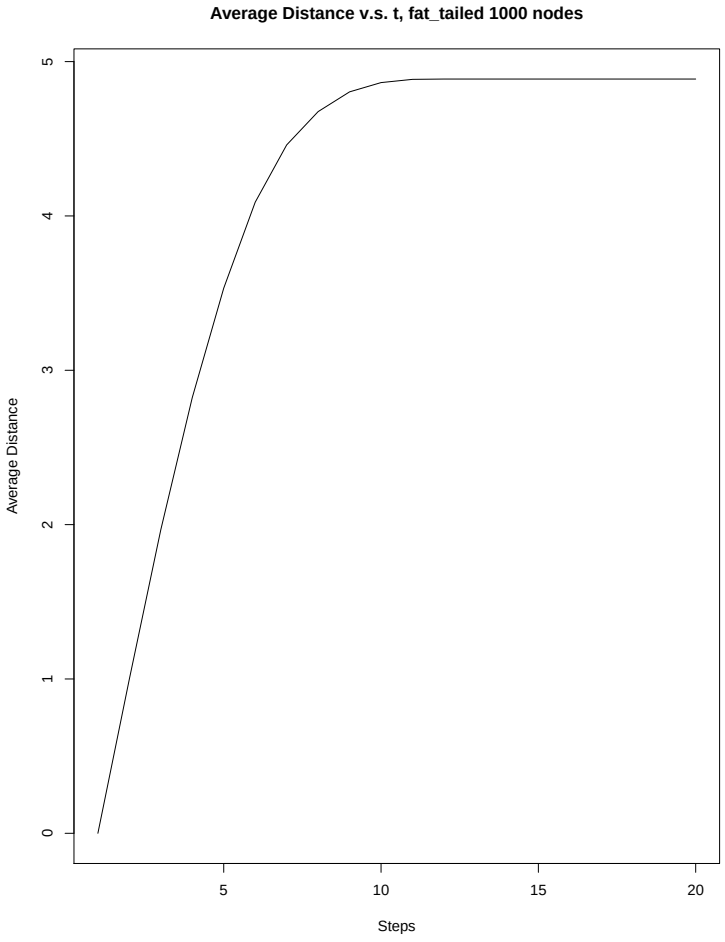
Comparing to 1000 nodes, which has about 0.5 of average standard deviation, the overall average standard deviation of 10000 nodes is smaller, which is only about 0.25 when the step is larger than 4.

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

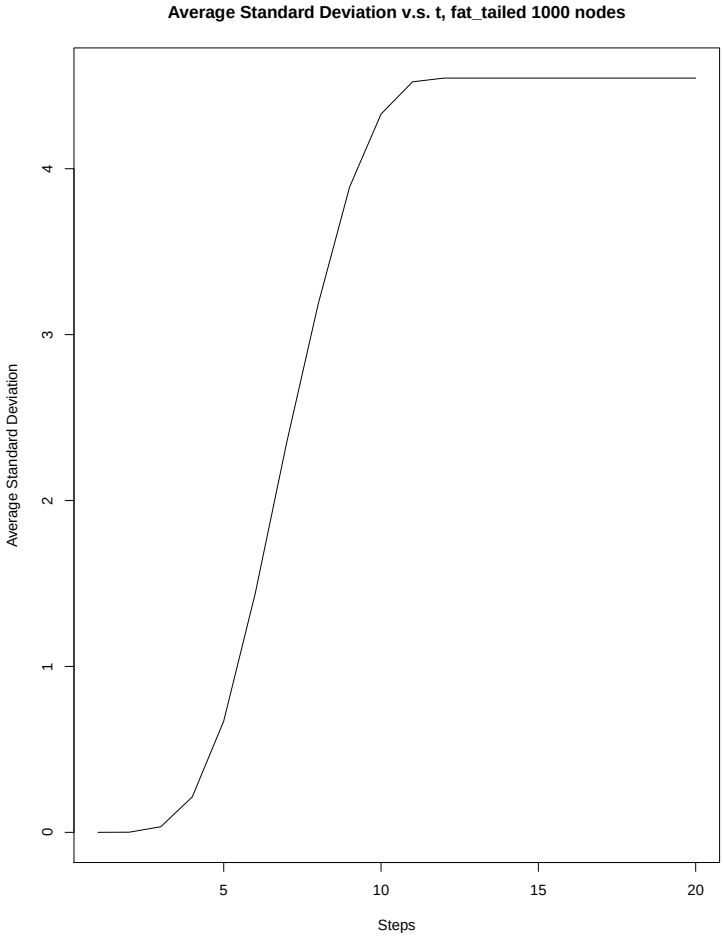
(a) Generate an undirected preferential attachment network with 1000 nodes, where each new node attaches to m = 1 old nodes.

We used *barabasi.game* in the *igraph* package to create the network.

(b) Let a random walker start from a randomly selected node. Measure and plot $s(t)$ v.s. t and $\sigma^2(t)$ v.s. t.

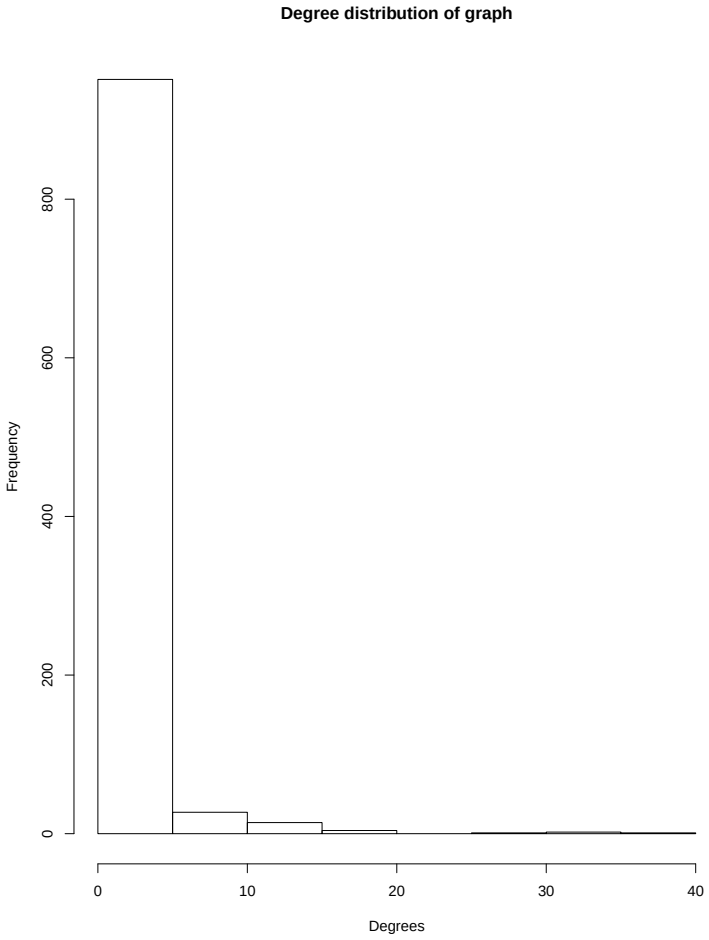


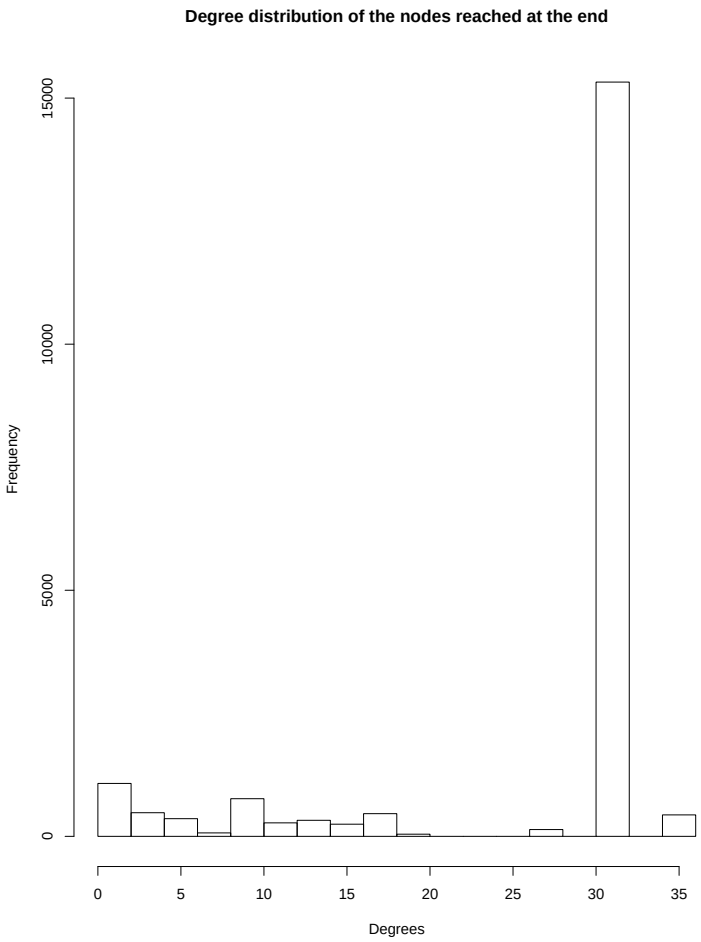
The average distance increases as t increases when t is smaller than 10 and becomes steady after the step is larger than 10. The average distance is around 5 when the step is larger than 10.



The average standard deviation increases as t increases when t is smaller than 11. Then the average standard deviation is 4.5 and becomes steady when step is larger than 11.

(c) Measure the degree distribution of the nodes reached at the end of the random walk on this network. How does it compare with the degree distribution of the graph?

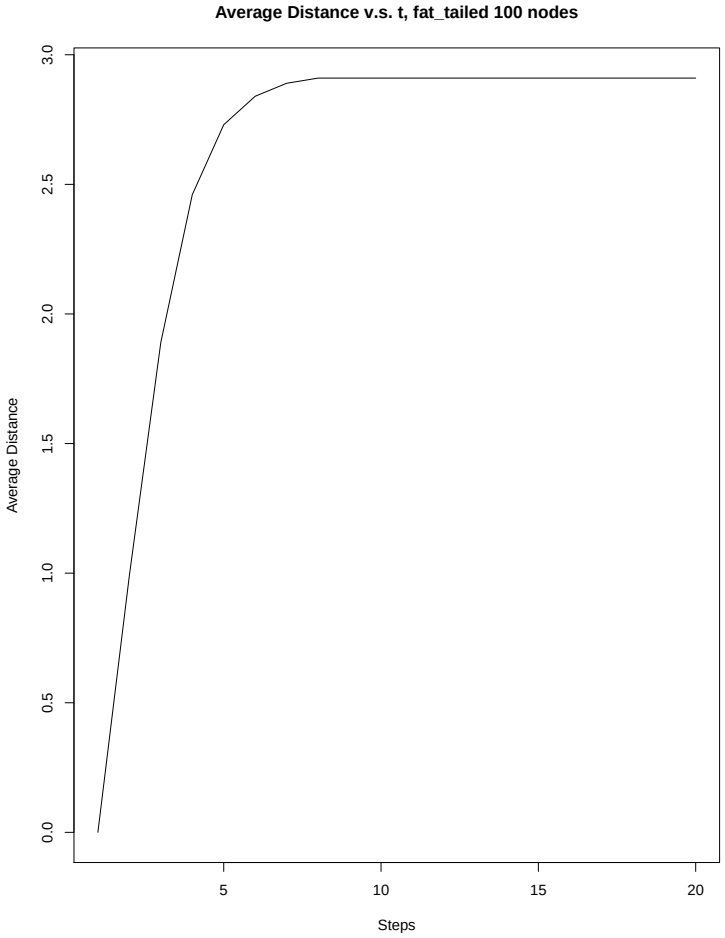




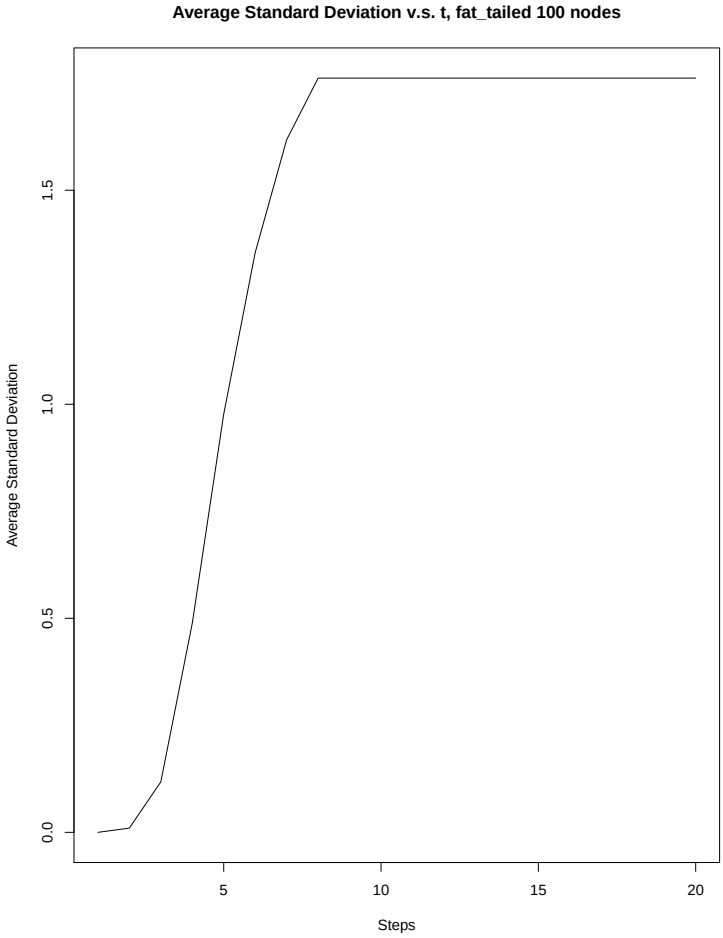
Regarding to the degree distribution of graph, most of the nodes have a number of degrees between 5 to 10. For the degree distribution of the nodes reaches at the end, most of them have number of degree between 30 and 35. This is because when performing random walk, we have high probability to be stuck at the first node.

(d) Repeat (b) for preferential attachment networks with 100 and 10000 nodes, and m = 1. Compare the results and explain qualitatively. Does the diameter of the network play a role?

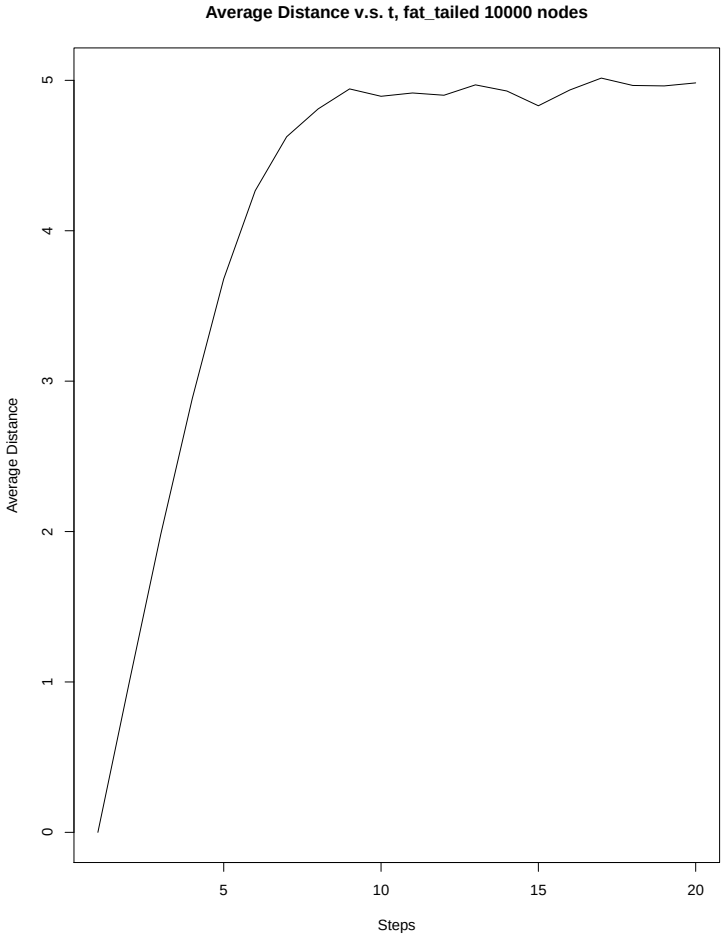
Though different numbers of nodes have different average distance and average standard deviation, they all have similar trends.



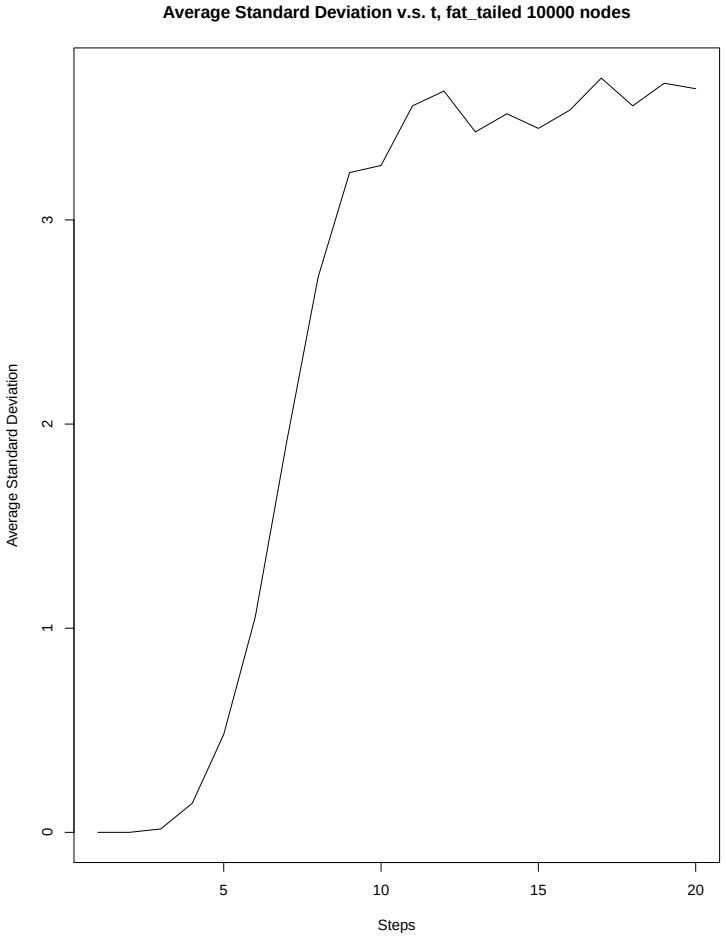
The average distance increases as t increases when t is smaller than 5 and becomes steady after the step is larger than 5. The average distance is around 3 when the step is larger than 5. The trend of it is similar to 1000 nodes but 1000 nodes has average distance around 5.



The average standard deviation increases as t increases when t is smaller than 7. Then the average standard deviation is 1.75 and becomes steady when step is larger than 7. The trend of it is similar to 1000 nodes but 1000 nodes has higher average standard deviation around 4.5.



The average distance increases as t increases when t is smaller than 7 and becomes steady after the step is larger than 7. The average distance is around 5 when the step is larger than 7. The trend of it is similar to 1000 nodes and 1000 nodes has average distance around 5.

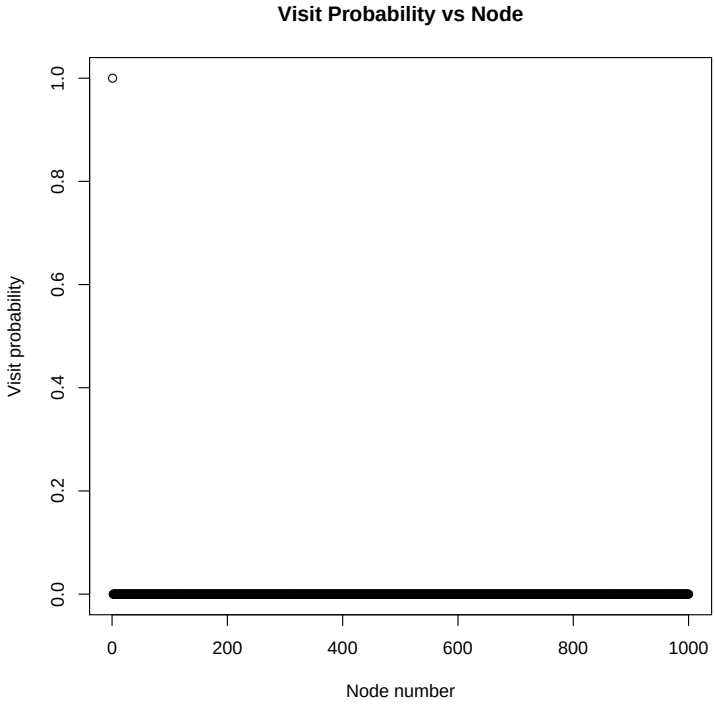


The average standard deviation increases as t increases when t is smaller than 10. Then the average standard deviation is 4 and becomes steady when step is larger than 10. The trend of it is similar to 1000 nodes but 1000 nodes has higher average standard deviation around 4.5.

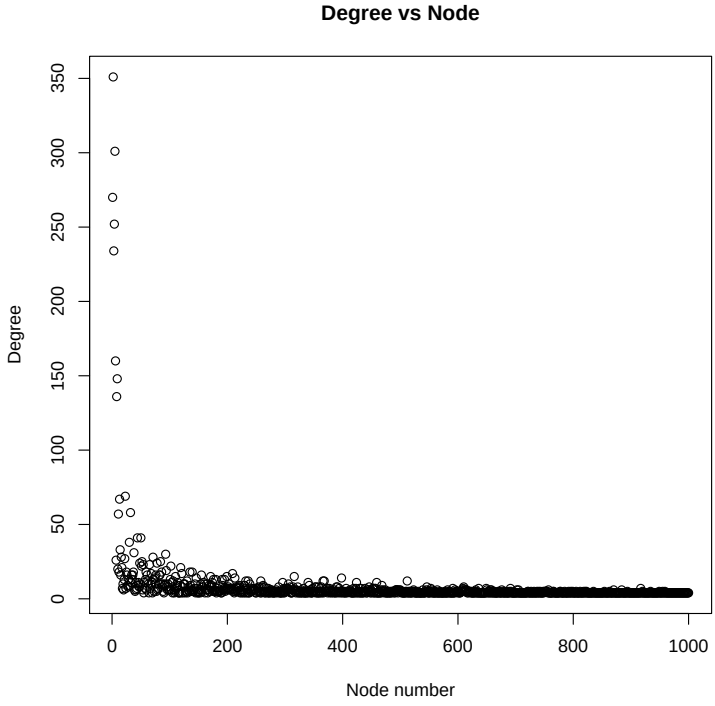
3. PageRank

(a) Create a directed random network with 1000 nodes, using the preferential attachment model, where m = 4. Note that in this directed model, the out-degree of every node is m, while the in-degrees follow a power law distribution. Measure the probability that the walker visits each node. Is this probability related to the degree of the nodes?

We used *barabasi.game* in the *igraph* package to create the specified network.
The probability that a random walker visits each node at steady state from 1000 different starting node is shown below. To ensure steady state is reached, we experimented by increasing the number of steps until the probability distribution no longer changed. Here we choose the number of steps to be 50.



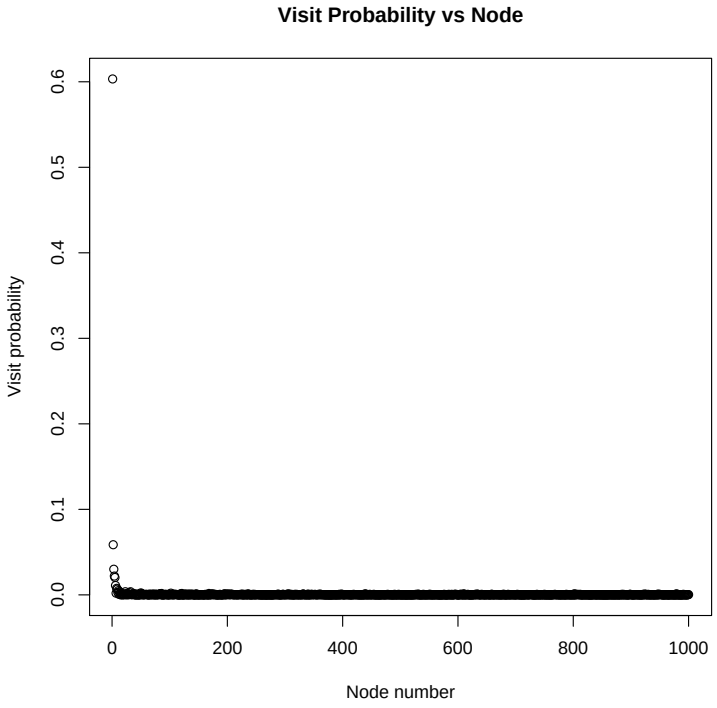
Below we show the degree of each node versus the node number. We see that the degree of each node decreases as the node number increases as we would expect for a preferential attachment network.



From the first figure above, we can see that the first node has a visit probability of 1 at steady state and the rest of the nodes are zero. The visit probability of each node at steady state is certainly related to the degree of each node. The situation can be rationalized as follows: The visit probability of each node of the random walker at each step is proportional the the in-degree of each node, so the first node has the highest probability of being visited. Therefore, given enough steps, the random walker from any starting node will definitely visit the first node. However, the first node in the directed preferential attachment network does not have out edges and therefore once the random walker visits it, it will remain at the first node. Thus, for large enough steps the random walker will be at the first node.

(b) In all previous questions, we didnt have any teleportation. Now, we use a teleportation probability of $\alpha = 0.15$. By performing random walks on the network created in 3(a), measure the probability that the walker visits each node. Is this probability related to the degree of the node?

The probability that a random walker visits each node at steady state with a teleportation probability of $\alpha = 0.15$ is shown below.

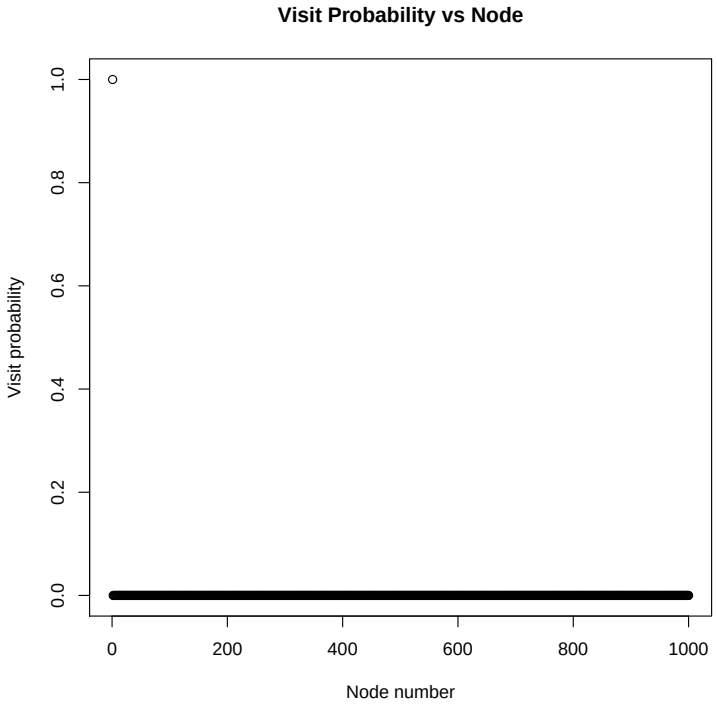


In this case, when the random walker visit the first node it will no longer being 'stuck' forever, but will have a probability of 0.15 to visit any other node at any step. Therefore the visit probability of the first node at steady state decreases compare to part (a) and the effect of teleportation is to increase the probability of other nodes to be visited at steady state. Furthermore, nodes with higher degree will still have a higher probability of being visited and this is why we observe the visit probability of the first few nodes (except the first) have some clear increases in probability compare to the rest and therefore the visit probability is related to the degree of the node.

4.Personalized PageRank

(a) Suppose you have your own notion of importance. Your interest in a node is proportional to the nodes PageRank, because you totally rely upon Google to decide which website to visit (assume that these nodes represent websites). Again, use random walk on network generated in part 3 to simulate this personalized PageRank. Here the teleportation probability to each node is proportional to its PageRank (as opposed to the regular PageRank, where at teleportation, the chance of visiting all nodes are the same and equal to 1/N). Again, let the teleportation probability be equal to $\alpha = 0.15$. Compare the results with 3(b).

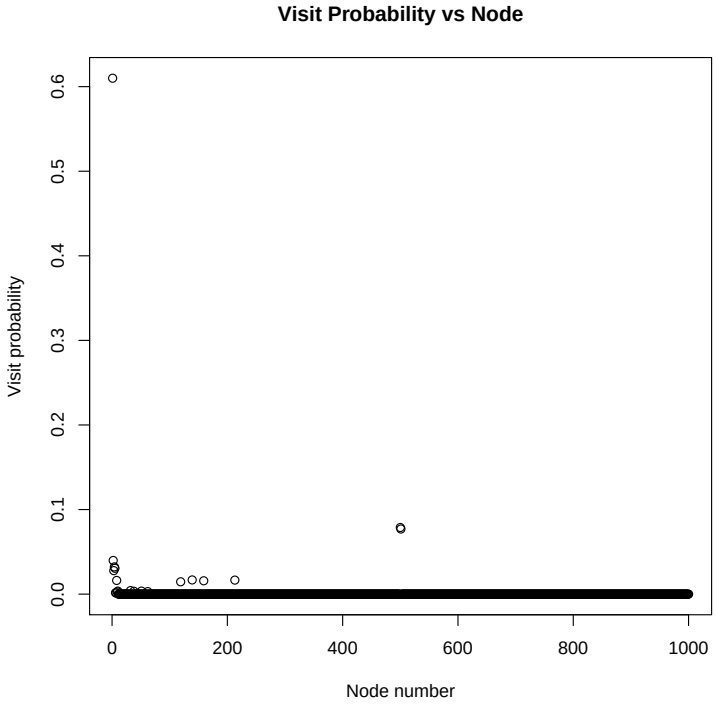
Here the node's PageRank is the visit probability we obtained from 3(a), which we simulated using random walk. The PageRank is 1 for the first node and 0 for the rest.



In 3(b), teleportation has equal probability to visit all nodes, so the visit probability increase for nodes with larger degree. However in this case, teleportation will only lead the random walker to the first node and the random walker will also visit the first node for large enough steps, thus the visit probability is still 1 for the first node and 0 for the rest. It is as if teleportation with PageRank will only lead the random walker to the node that it will eventually visit. Thus we did not observe the visit probabilities of the first few nodes increase compare to 3(b).

(b) Find two nodes in the network with median PageRanks. Repeat part (a) if teleportations land only on those two nodes (with probabilities 1/2, 1/2). How are the PageRank values affected?

Here we pick node 500 and 501 to have a probability of 1/2. The probability that a random walker visits each node at steady state is shown below.



In this case, the visit probability for the first node decreases as expected because teleportation leads the random walker to node 500 and 501, which has obvious increase in visit probability. In fact, they have visit probability approaching 0.075 as one would've expected. In our generated network, node 500 has directed edges connected to nodes 1, 139, 159, 213 and node 501 has directed edges connected to 4, 5, 8, 119. We can see that these nodes that is directly connected to node 500 and 501 have clear increase in visit probability. Furthermore, the first few nodes with large in-degree also have slight increase in visit probability as in the case in 3(b).

(c) More or less, (b) is what happens in the real world, in that a user browsing the web only teleports to a set of trusted web pages. However, this is against the different assumption of normal PageRank, where we assume that peoples interest in all nodes are the same. Can you take into account the effect of this self-reinforcement and adjust the PageRank equation?

The normal PageRank equation is

$$PR(u) = \frac{\alpha}{N} + (1-\alpha) \sum_{v \in B_u} \frac{PR(v)}{L(v)} \tag{3}$$

where $PR(u)$ is the PageRank of node u, α is the teleportation probability, $1/N$ is the probability of visiting the node via teleportation, B_u is the subset of nodes with edges directed toward node u, and $L(v)$ is the total out-degree of node v. This is the situation given in 3(b) and this equation works for all nodes in that case.
Now for 4(b), the probability of visiting the node via teleportation is only non-zero and equal to $1/2$ for two of the nodes we picked. For these two nodes, the PageRank equation should be modified to

$$PR(u) = \frac{\alpha}{2} + (1-\alpha) \sum_{v \in B_u} \frac{PR(v)}{L(v)} \tag{4}$$

and for the rest of the nodes with zero the probability of visiting the node via teleportation, the PageRank equation becomes

$$PR(u) = (1 - \alpha) \sum_{v \in B_u} \frac{PR(v)}{L(v)} \tag{5}$$