EE232 Project 1
Random Graphs and Random Walks

Team members:
Sonali Garg (104944076)
Aashna Agarwal (404943216)
Shweta Sood(905029230)
Karan Sanwal(205028682)
Ashish Shah(804946005)

Part 1: Generating Random Networks

Q1. Create random networks using Erdös-Rényi (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.

In this part we create graphs using the 'erdos.renyi.game()' function using n, number of nodes and p, probability for drawing an edge between two arbitrary vertices. We find the distribution using the 'degree' function and plot histograms for each of the p.

The below plots show the degree distribution for graphs generated with different p, where p is the probability with which each edge is added between a pair of nodes. For each node there can be (n-1) neighbours out of which we select the edges with probability of p. So if the degree of the node is k, then the degree distribution will hence follow the below form.

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

The above expression is of a binomial distribution. The below plots of the randomly shown graphs show the distribution similar to Binomial Distribution.

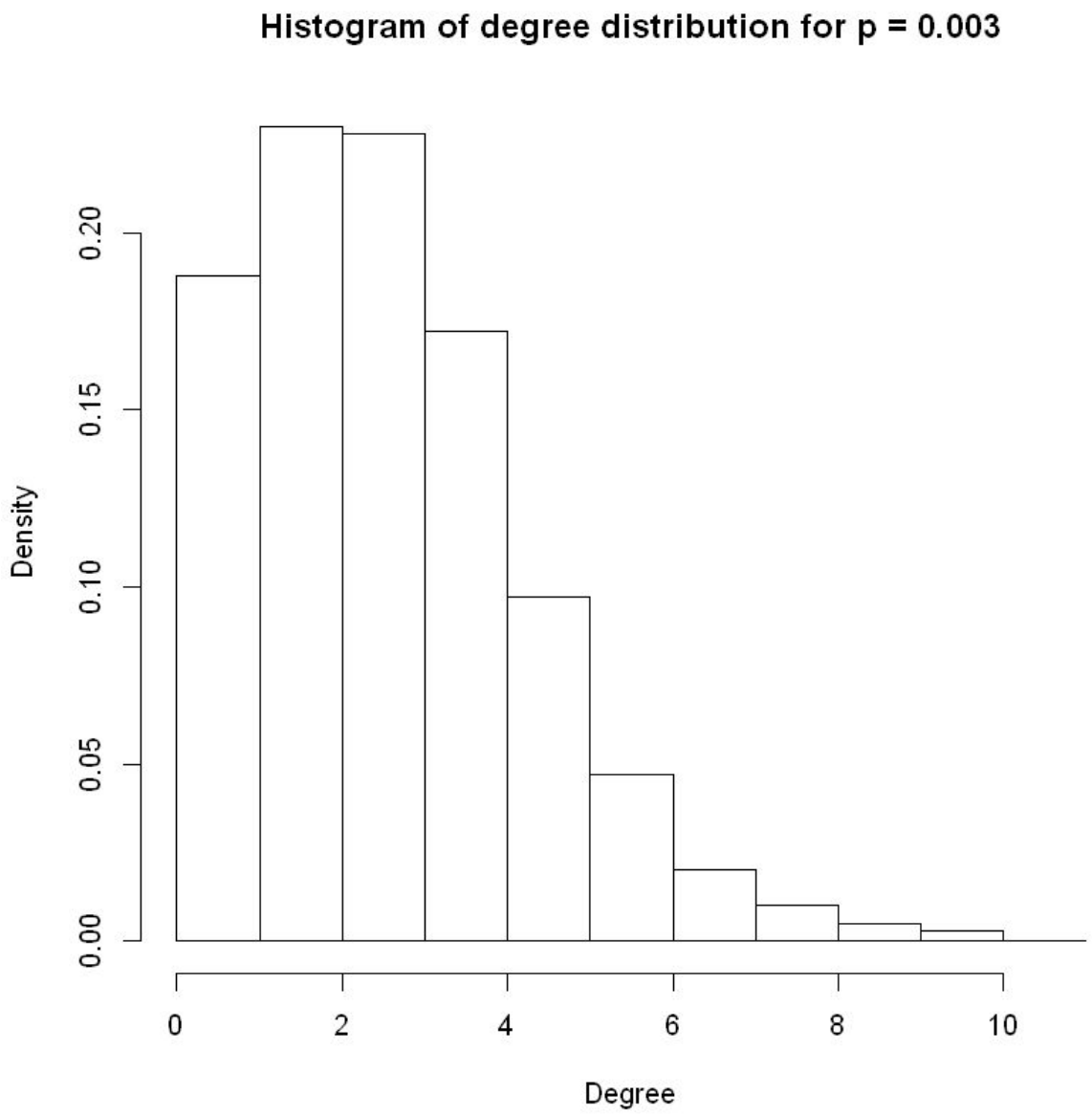


Fig 1: Histogram of degree distribution for p=0.003

Mean of the above degree distribution: 2.9966
Variance of the above degree distribution: 3.01725861258613

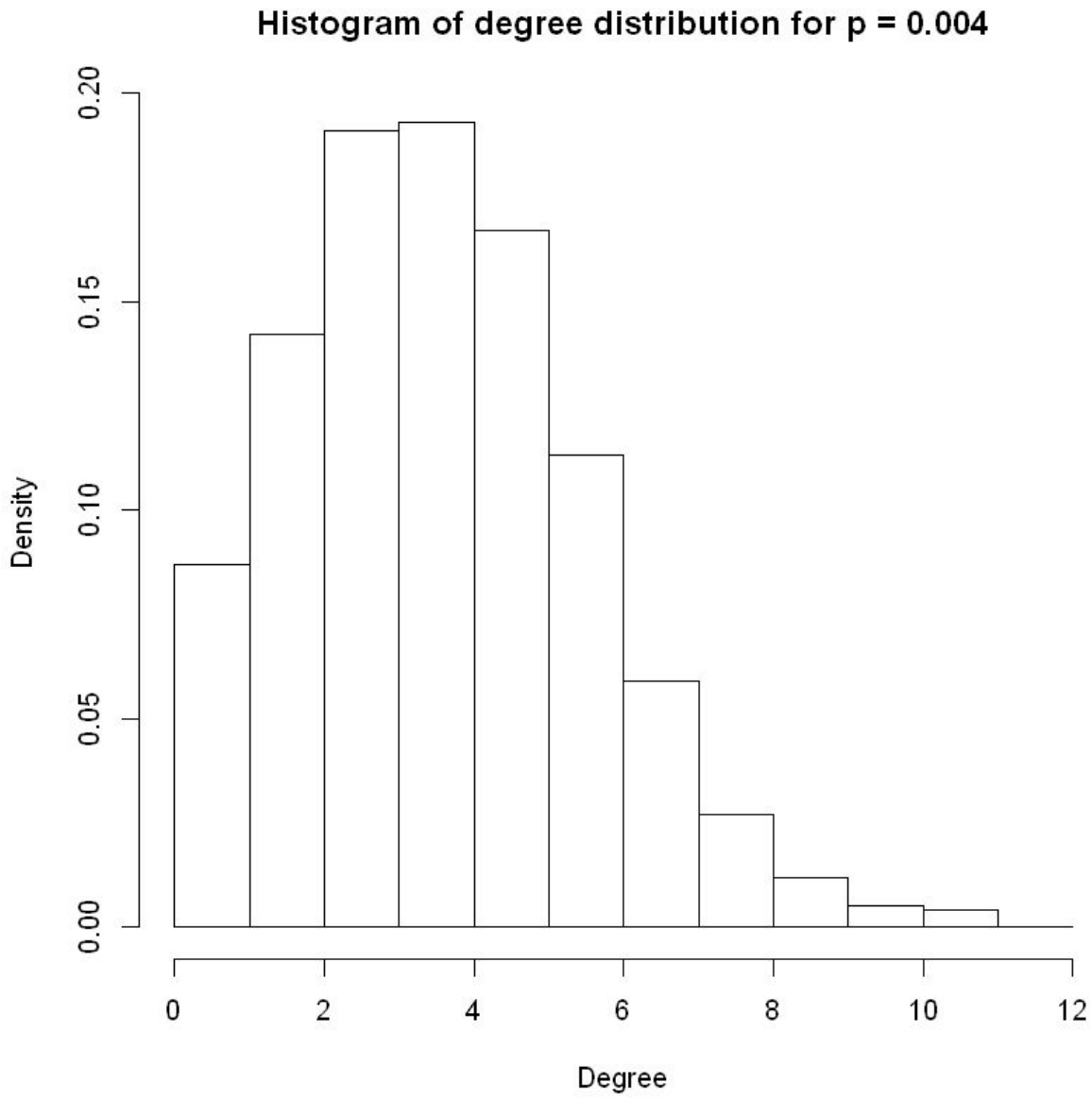


Fig 2: Histogram of degree distribution for p=0.004

Mean of the above degree distribution: 4.00364
Variance of the above degree distribution: 3.95054625586256

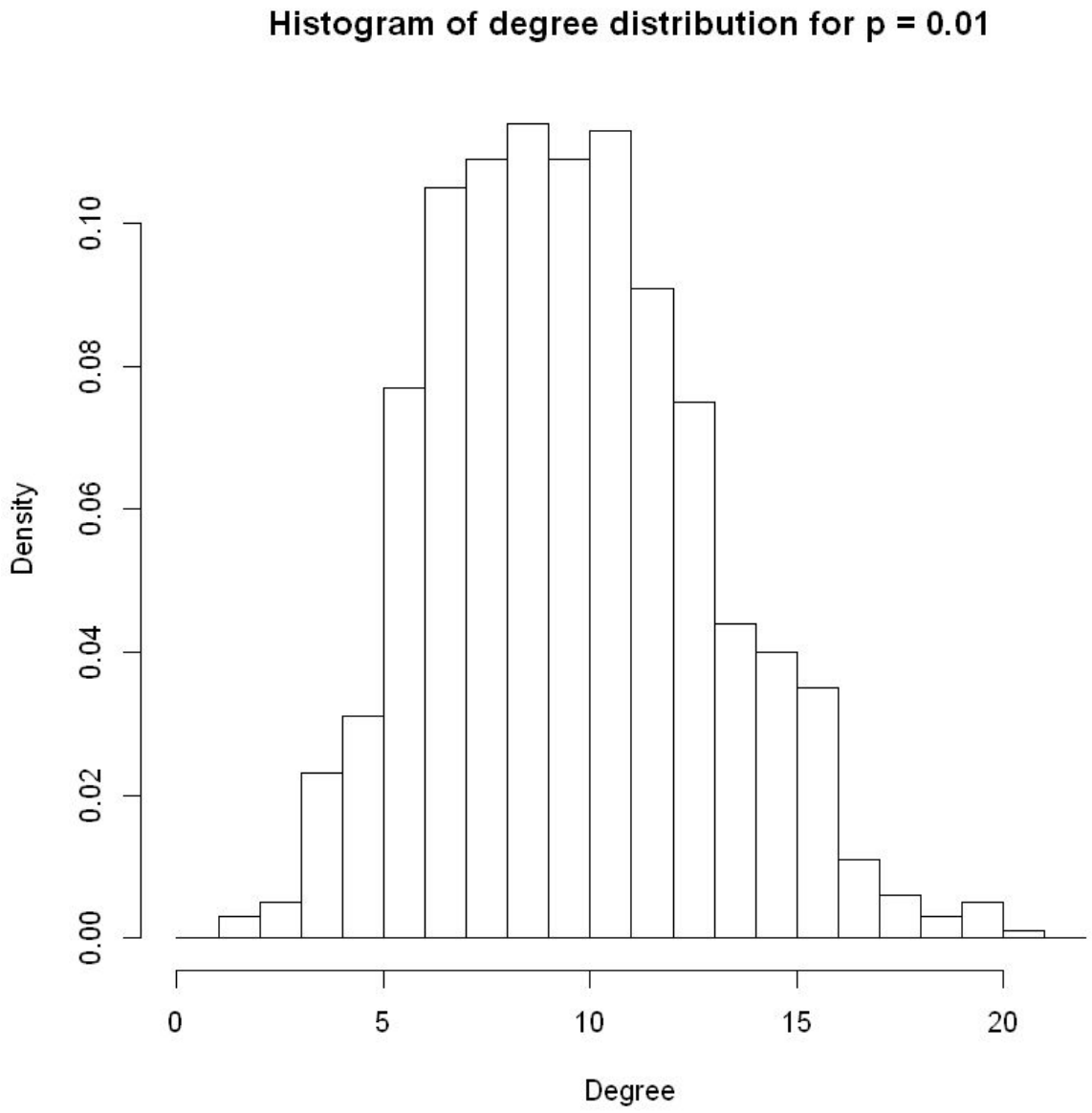


Fig 3: Histogram of degree distribution for p=0.01

Mean of the above degree distribution: 9.97416
Variance of the above degree distribution: 9.86793097370974

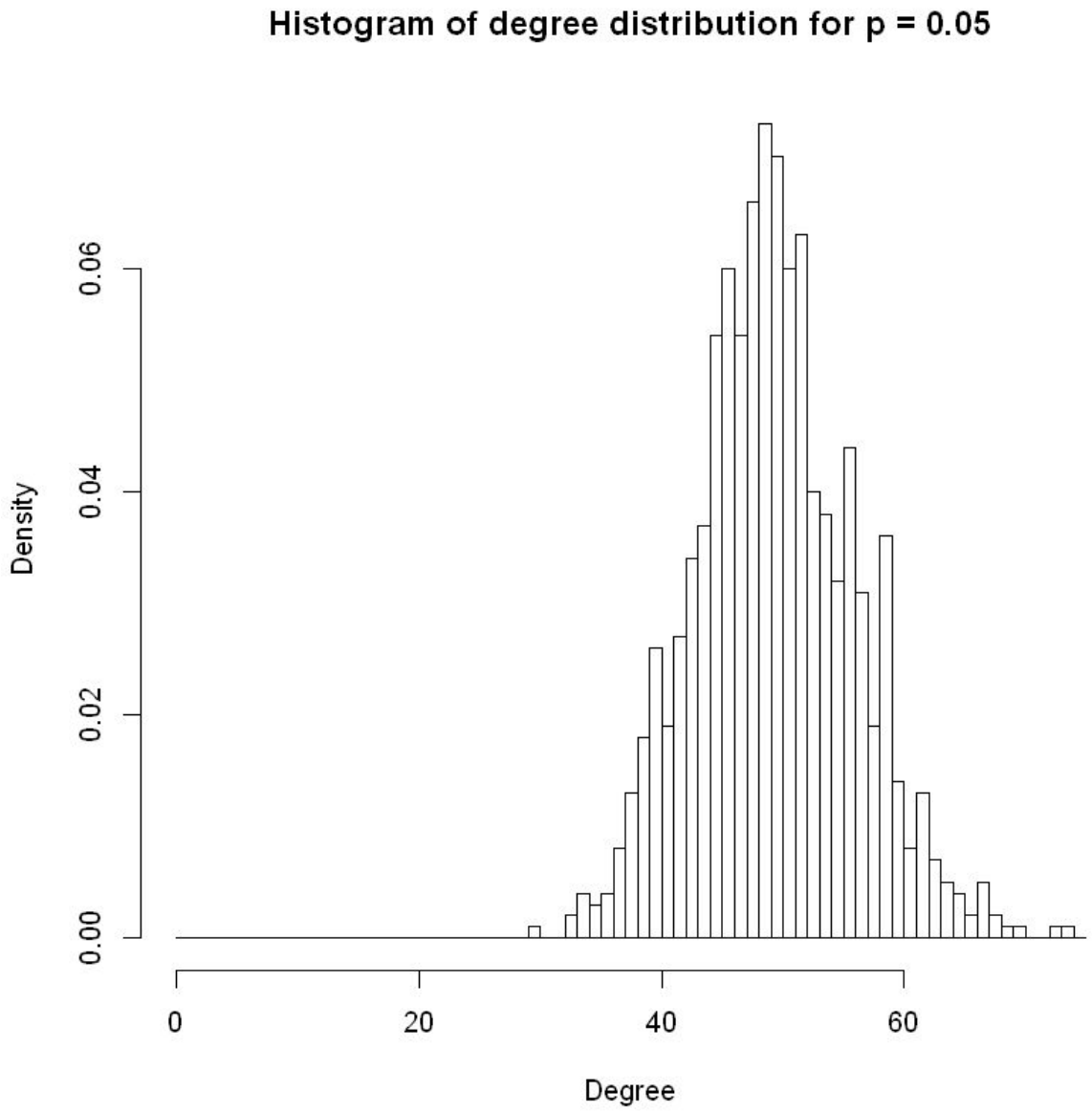


Fig 4: Histogram of degree distribution for p=0.05

Mean of the above degree distribution: 49.93958
Variance of the above degree distribution: 47.6728661522615

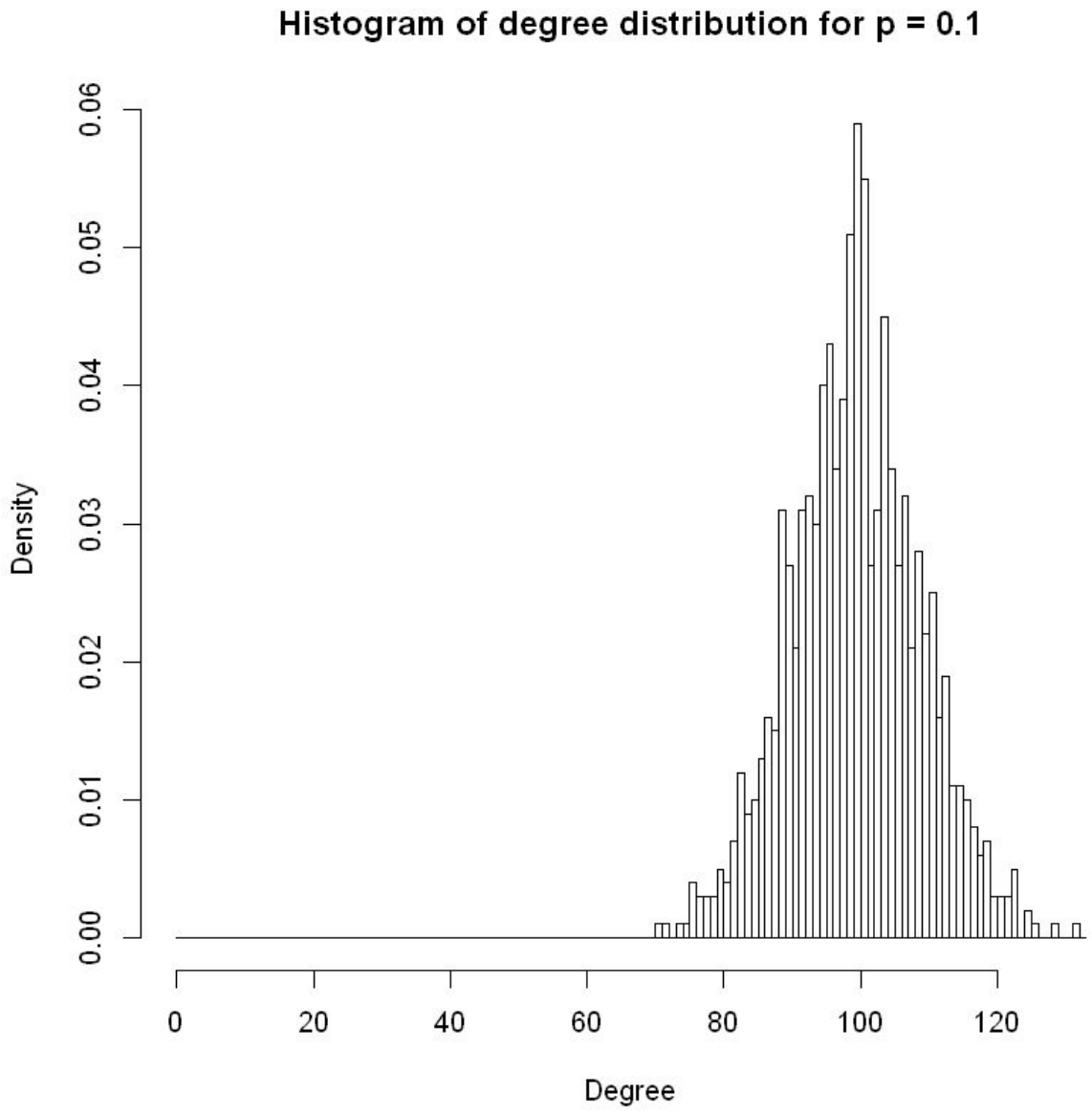


Fig 5: Histogram of degree distribution for p=0.1

Mean of the above degree distribution: 99.82844
Variance of the above degree distribution: 89.7669848362483

The distribution observed is a binomial distribution.
We have shown above why it is a binomial distribution.
Theoretically, Mean is (n-1)*p and Variance is (n-1)p(1-p).

For the above graphs, we theoretically see the mean and variance to be the following.
For graph 1, mean is 2.997 and variance is 2.988

For graph 2, mean is 3.996 and variance is 3.98
For graph 3, mean is 9.99 and variance is 9.89
For graph 4, mean is 49.95 and variance is 47.45
For graph 5, mean is 99.9 and variance is 89.91

We see the values of mean and variance obtained both via code and theoretically match with each other and agree with each other.
This shows us that the graphs built using n and p follow the Erdos Renyi model and have the expected mean and variance.

(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 all the random realizations aren't connected.
The greatest connected component of the graph is the connected component with the largest number of nodes.
A value pc is defined such that the generated random networks are disconnected when p<pc and connected when p>pc. Using is.connected() method we checked if the graph is connected or disconnected each time increasing the value of p by 0.001, starting from 0.000. We recorded 100 different observations for 100 such random graphs and took the average to get accurate results.
Threshold Probability = pc = 0.00762

We can also analytically derive the value of threshold probability (pc) using the Erdos-Renyi Asymptotic Expression, the random graph g, is disconnected if the link density p is below the connectivity threshold using the formula,
Pc = ln(n)/n
pc=ln(1000)/1000 = 0.0069
Thus it can be seen that the analytically value calculated is very close to the value computed through the program.

Graph 1: Is connected: FALSE
Graph 2: Is connected: FALSE

Graph 3: Is connected: TRUE
Diameter of connected component: 5.32

Graph 4: Is connected: TRUE
Diameter of connected component: 3

Graph 5: Is connected: TRUE
Diameter of connected component: 3

For Graph 1, we will find the GCC.
Vertex count of the GCC for Graph 1: 938
Diameter of the GCC for Graph 1: 13

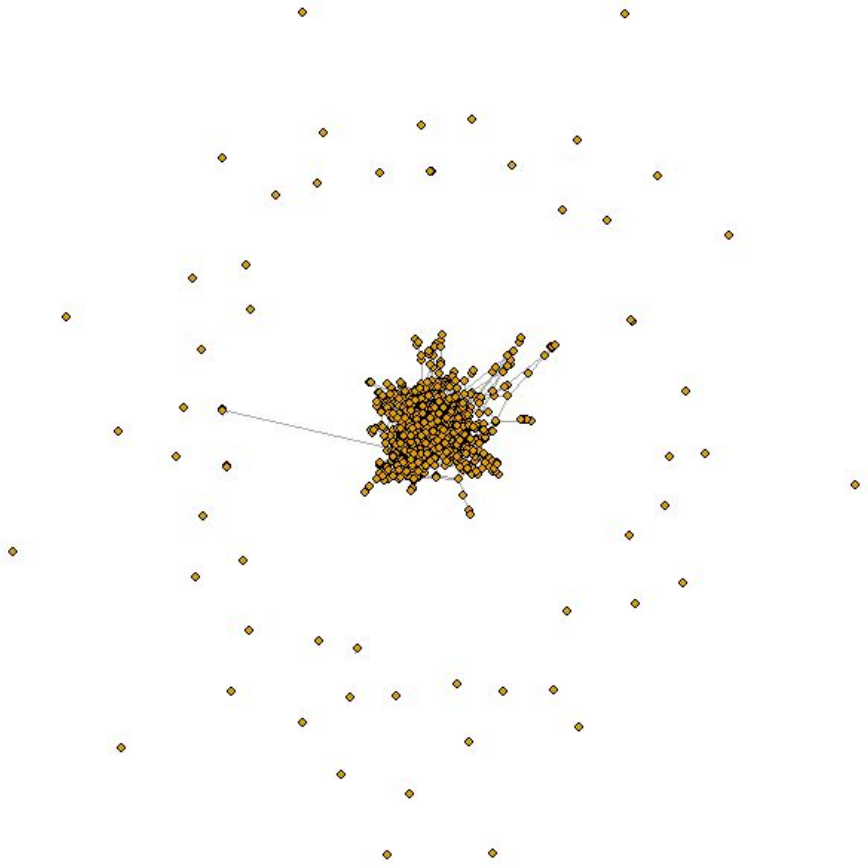


Fig 6: Unconnected Graph 1

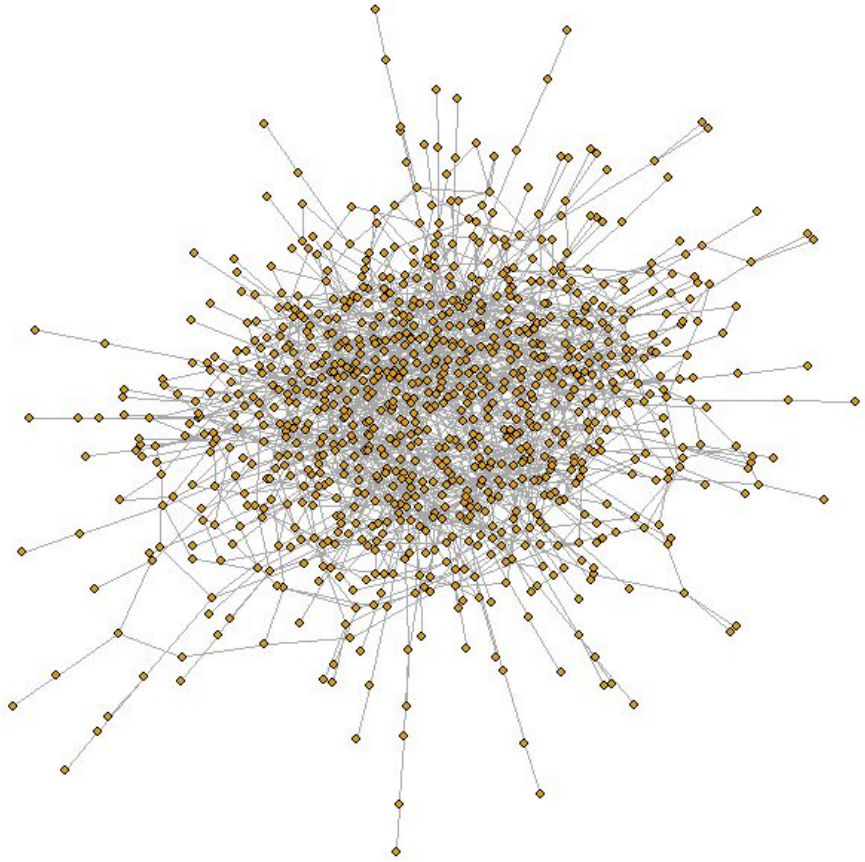
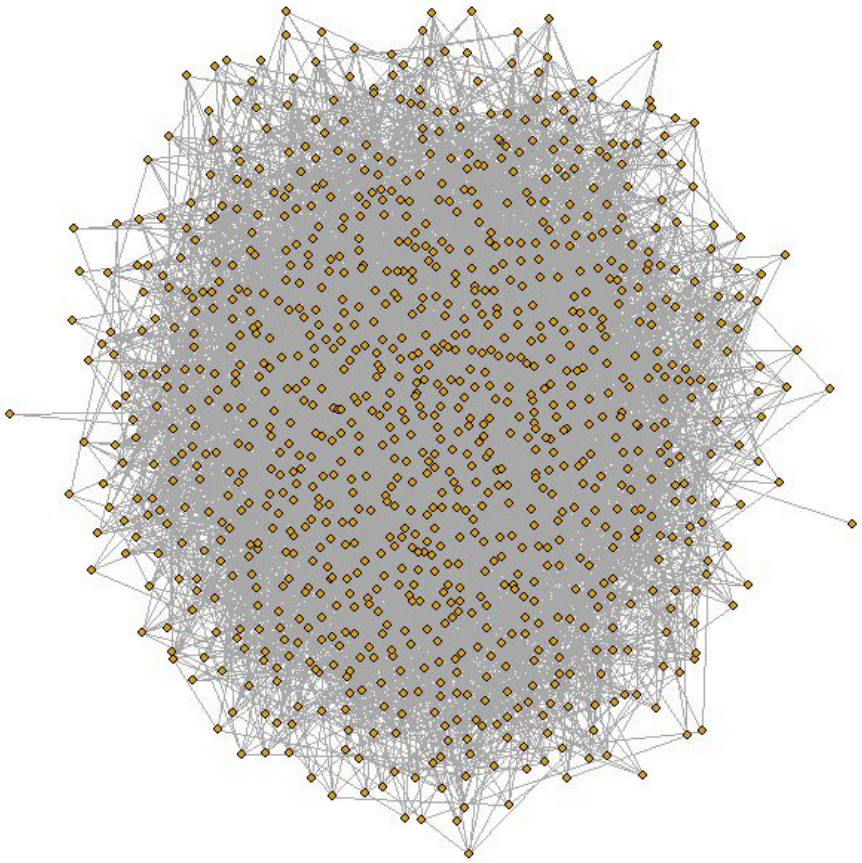


Fig 7: GCC of Graph 1

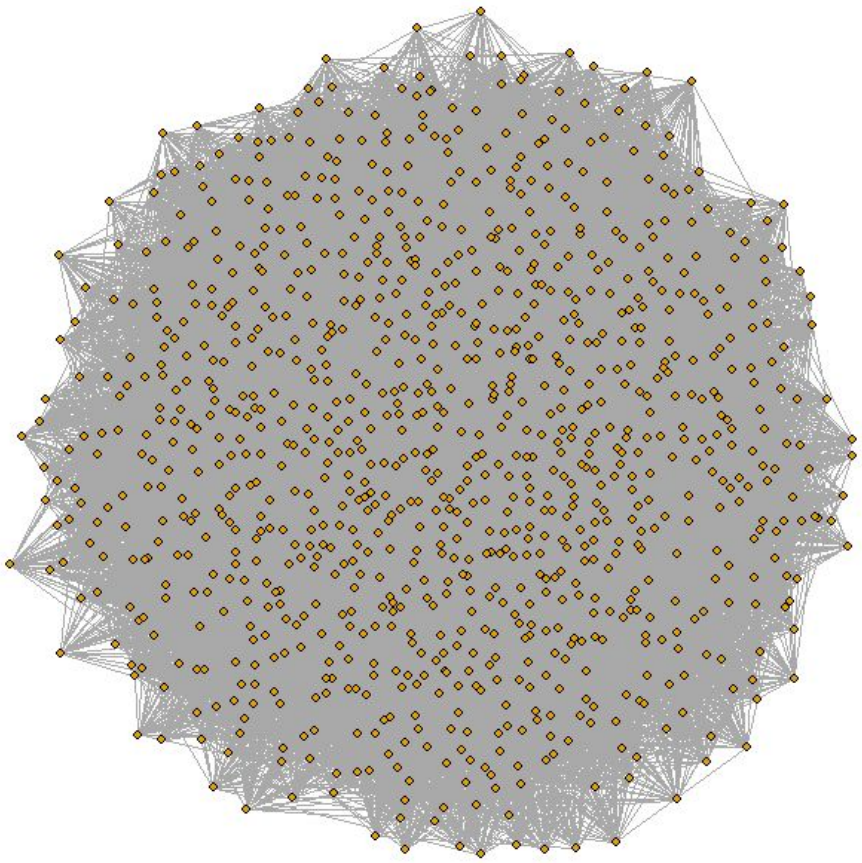
The probability that a generated network is connected is given by the formula c*ln(n)/n.
Here n is 1000 and c=1.
We see that this gives us the value 0.0069.
So if the p<0.0069 then the graph will not be connected and we will find the GCC for it. If p>0.0069, the graph will be connected and we can see this with graph 3,4 and 5.

For graph 2, it is also not connected.

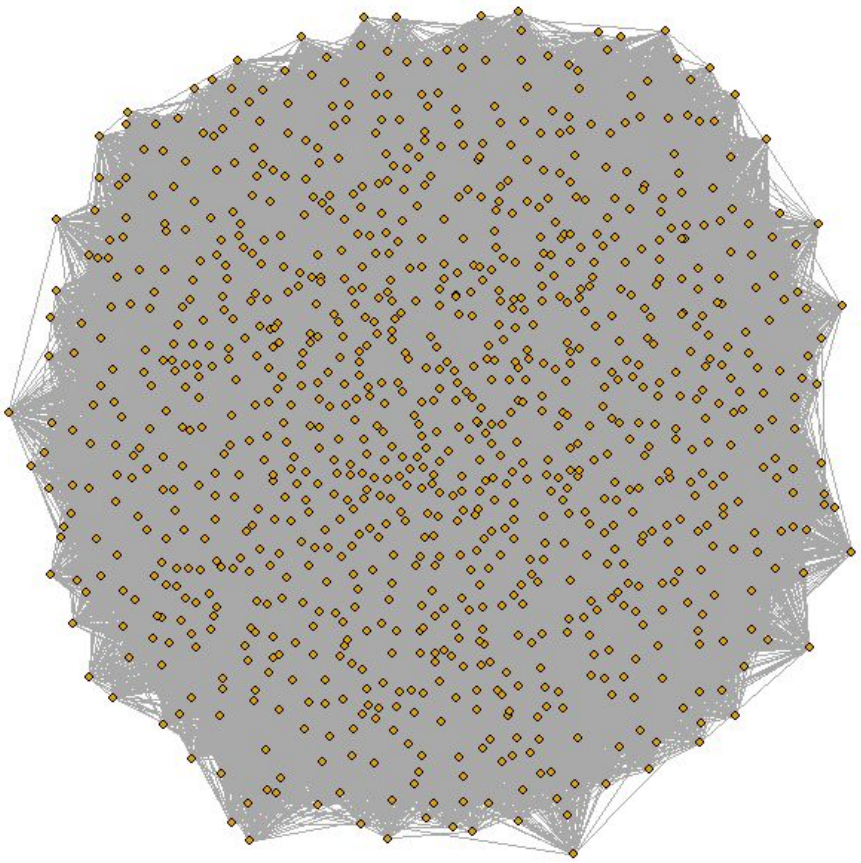
Graph 3 is connected.



Graph 4 is connected



Graph 5 is connected



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(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?

In this part, we first start the sweep from p=0.000 and go on till log(n)/n which is 0.0069 or ~0.007 incrementing at each step by 0.001. For each p, we created 100 random networks and found the normalized GCC for each.
We also went ahead and swept till 0.09 to make sure that the graphs become connected after 0.007. At probability 0.07, the graph becomes connected.

The size of the GCC(Greatest connected component with largest amount of nodes) is given by the following:
For a graph with n nodes and probability p, if p< (1-e)ln(n)/n then the graph will mostly not be connected. If p>(1-e)ln(n)/n then it'll be connected. Here n=1000 and through this equation we get the value of 0.007. We observe the graph is connected after 0.007. Any value of p above this will to a connected graph and any value below this will give us an unconnected graph.

We define point of emergence as the p value beyond which the size of GCC~n. Here we can see that the point of emergence is 0.01.

The theoretical values derived in lectures are as follows.
c*ln(n)/n where n=1000 and c=1.
Here we see that the value we get is 0.0069~0.007.
We have plotted the normalized gcc value with its corresponding p value.

Our criterion of emergence matches with the values derived in class.

From the scatter plot, we see that the size of GCC varies much when the p is smaller and the variance keeps on decreasing. This happens when p is low the graph will be sparse because based on the random initializations, the connected parts of the graph can vary. The variation reduces as the graph becomes dense and p increases.

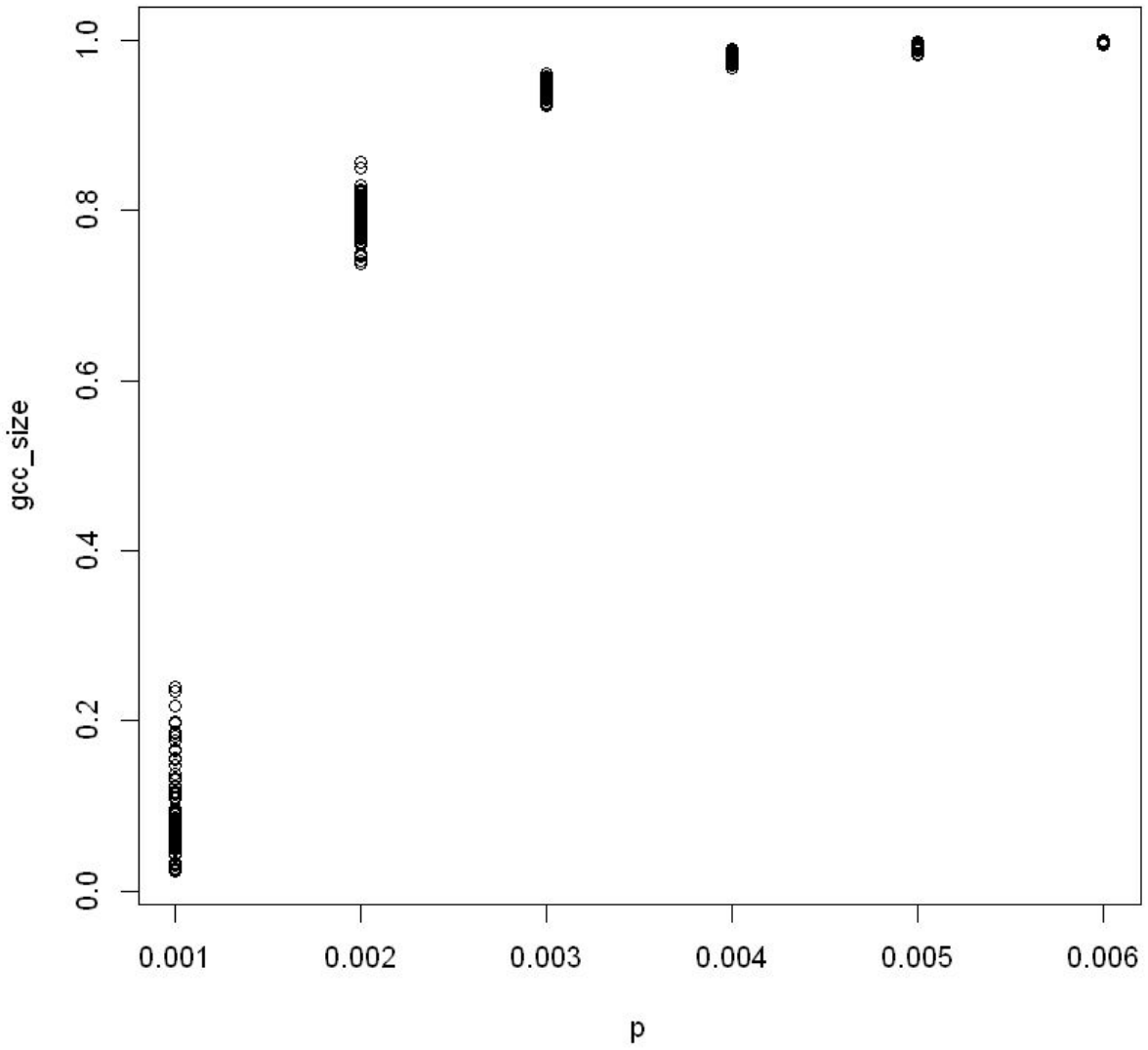


Fig: Plot of normalized GCC size vs p

(d) i. 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?

ii. Repeat the same for c = 1.

iii. 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.

i. In this part, we sweep over the number of nodes ‘n’ from 100 to 10000. Here c=0.5. We know p=c/n. For each n we find the p and create a graph for it.
We plot the expected size of GCC with n.

We observe that for c=0.5 the graph is more spread out and for each n, the GCC lies in a range of 5-25. It is more sparse than the others we’ll see below.
But as we increase c to 1, we see that the graph is now more dense and climbing towards a peak. The GCC varies from 0-800 mostly. As c increases, the size of GCC increases. The graph increases exponentially.

Through this we see a trend being followed in all the graphs. As c increases we see that the size of GCC also increases (as expected). We see this as c=0.5, the size of gcc isn’t too big but as c increases, we see values of 1.1, 1.2, the size of gcc also increases.
With the third graph, we are able to see a clear picture between c=1.1,1.2 and 1.3. The rightmost(Red) is what we get for c=1.1. And the graph moves to the left and becomes more dense as we increase c. Through the formula c=n*p we know c is directly proportional to p. The points on the graph become more closer to each other and the graph comes closer to the y-axis.

For c=0.5

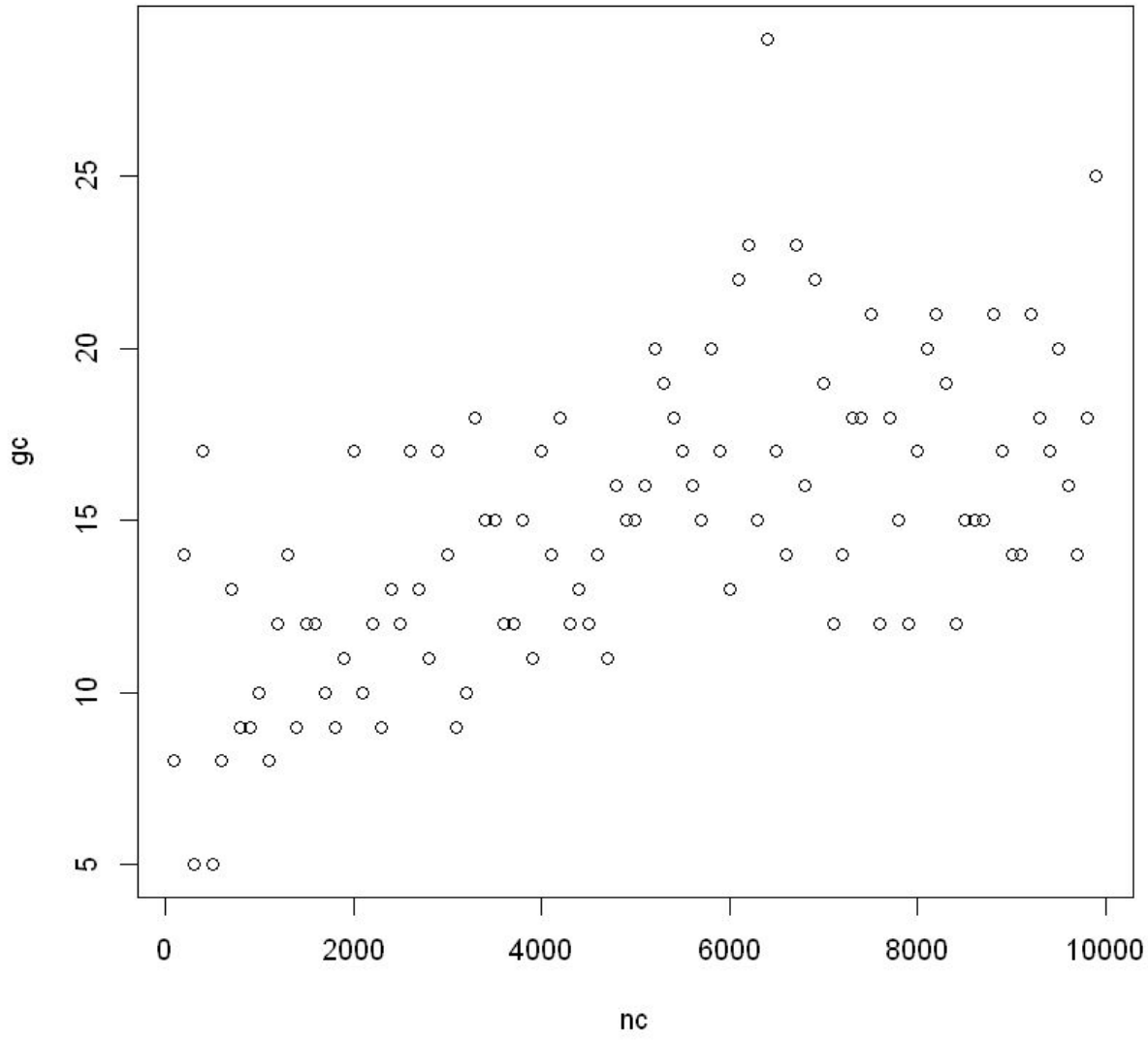


Fig: Plot of expected size of GCC as a function of n for c=0.5

ii. For c=1

Here we see a graph which is not as sparse as the one above but still is spaced out. It is following the exponential trend and is slowly becoming more dense and coming towards(nearer) the y-axis.

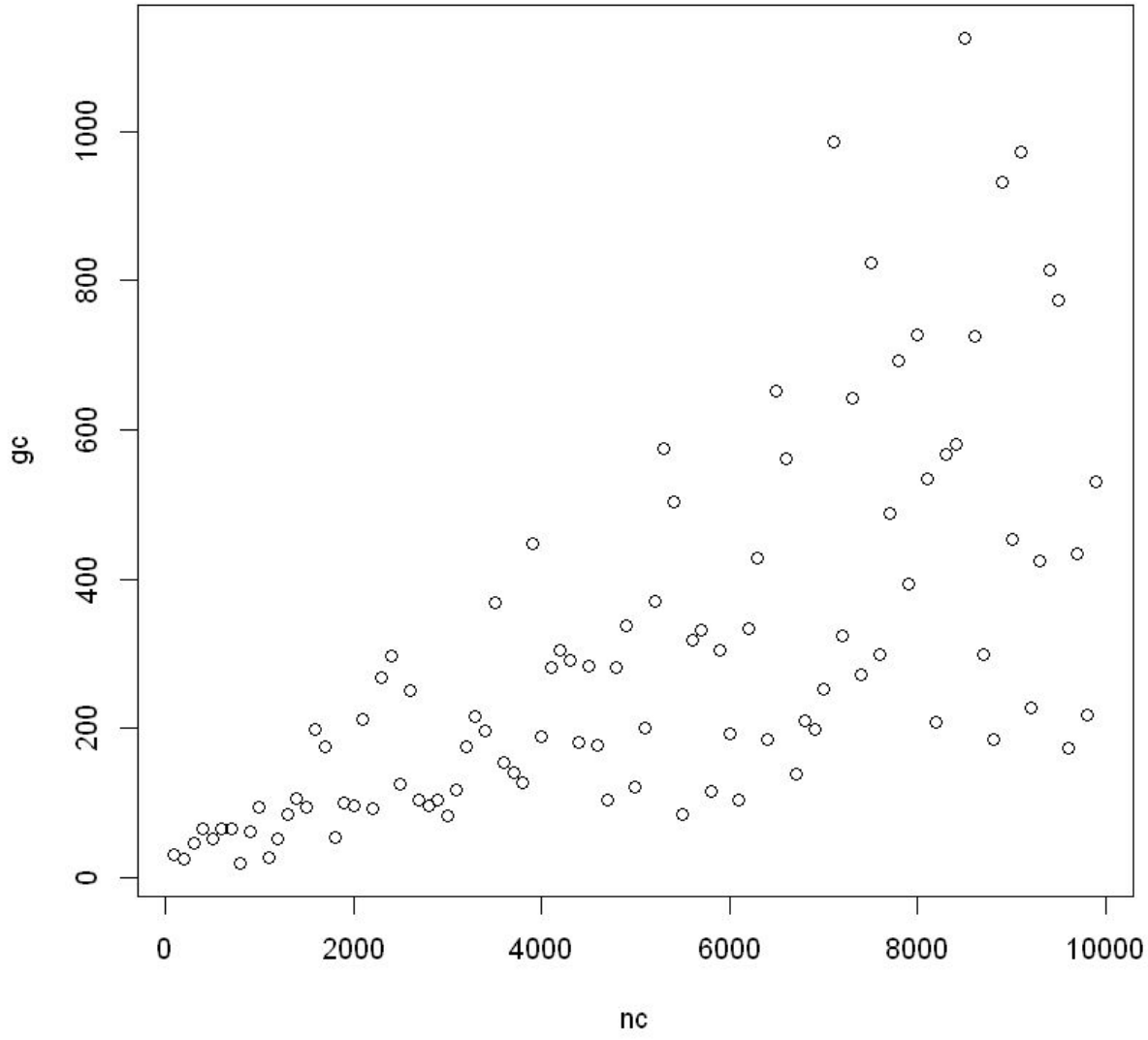
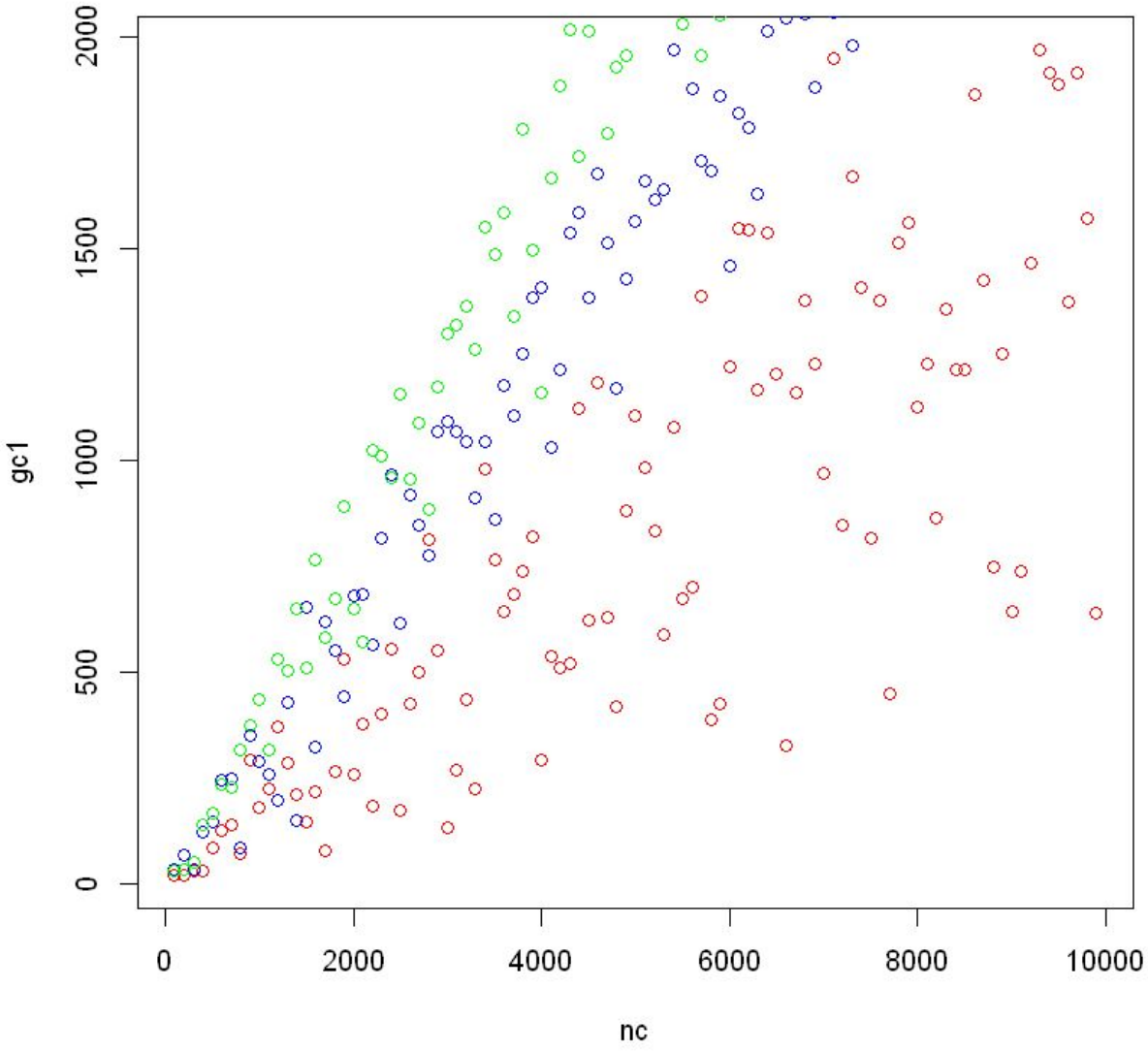


Fig: Plot of expected size of GCC as a function of n for c=1

iii. For c=1.1,1.2 and 1.3
Legend:
Red=c:1.1
Blue=c:1.2
Green=c:1.3
Here we can clearly see the trend which is being observed by the graph. This exponential trend can be seen in all three parts. Red is where c=1.1 and it is the furthest from y-axis. It is the most sparse out of the 3. Then we see ‘Blue’ which is c=1.2. It is more dense and towards the y axis. Here we see the points becoming much close to each other. Lastly we see ‘Green’ where c=1.3. Here the points are the closest and so is the graph towards the y-axis.



Q2 Create networks using preferential attachment model.

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

Solution:

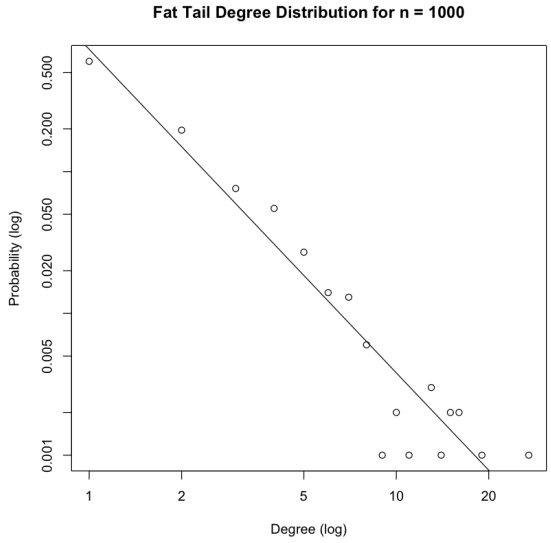
Yes, it is always connected.

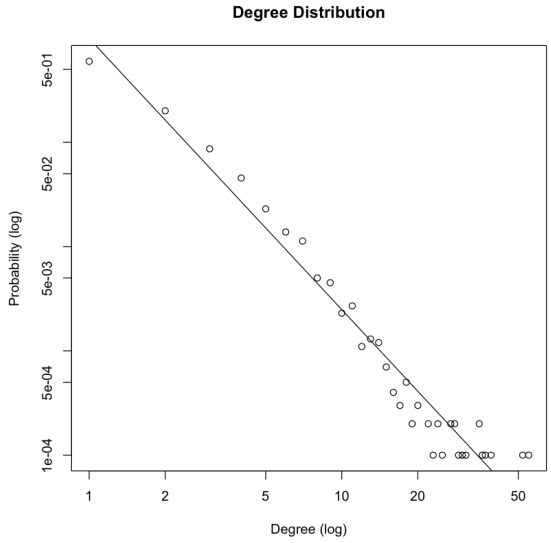
(b) Use fast greedy method to find the community structure. Measure modularity.

Solution:
The modularity is **0.932667402136873**

(c) Try to generate a larger network with 10000 nodes using the same model. Compute modularity. How is it compared to the smaller network's modularity?
Solution:
Modularity of larger network is higher, **0.978678015816396**. High modularity implies that the network is very well clustered (or partitioned) into communities due to preferential attachment of nodes to higher degree nodes.

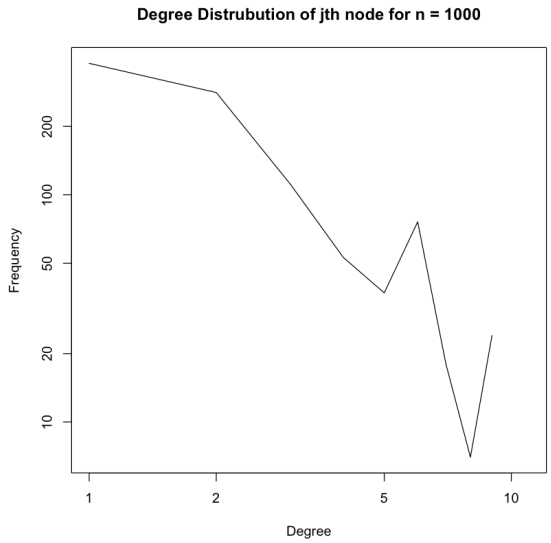
(d) Plot the degree distribution in a log-log scale for both n = 1000; 10000, then estimate the slope of the plot.
Solution:

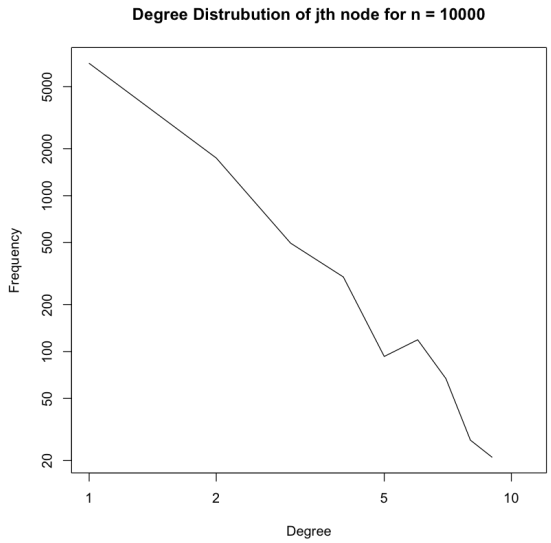




The slope of the plot with n = 1000 is **-2.2815**.
The slope of the plot with n = 10000 is **-2.600126**.

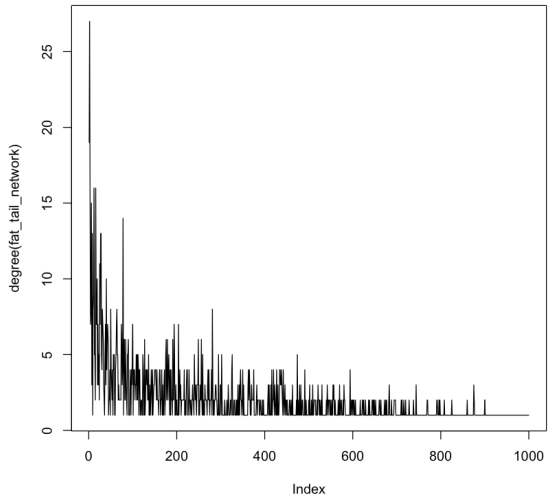
(e) randomly pick a node i, and then randomly pick a neighbor j. Plot the degree distribution of nodes j in the log-log scale. How does this differ from the node degree distribution?
Solution:

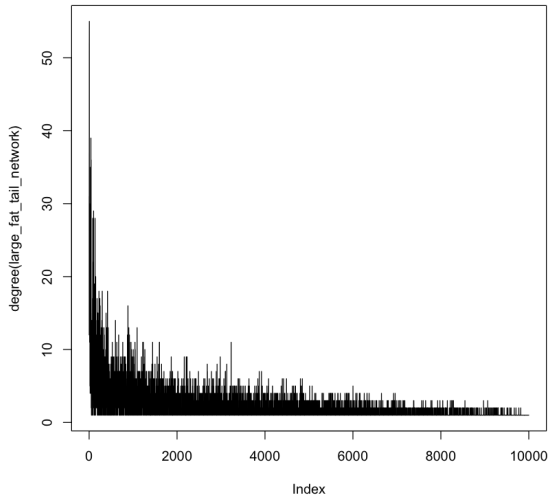




The node degree distribution exhibited power law perfectly, the above graph, although are in similar lines but doesn't fit with the power law distribution law well.

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





Here, we have considered age as index value, i.e. age has been considered as time-stamp. For example, the oldest node which got created at timestamp = 0, has been considered as age = 0.

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

For m=2:

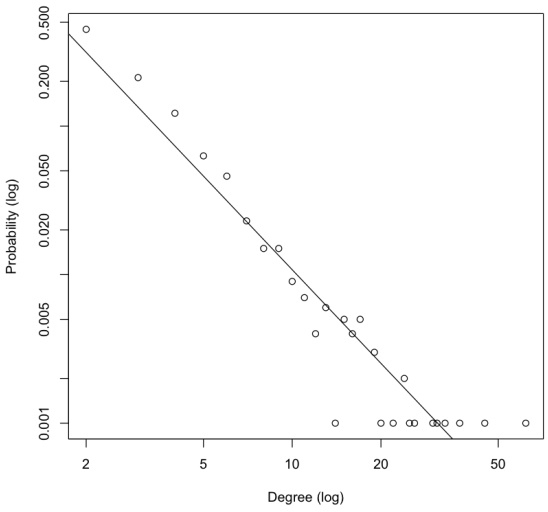
(a) Create an undirected network with n = 1000 nodes, with preferential attachment model, where each new node attaches to m = 2 old nodes. Is such a network always connected?
Solution:
Yes, it is always connected.

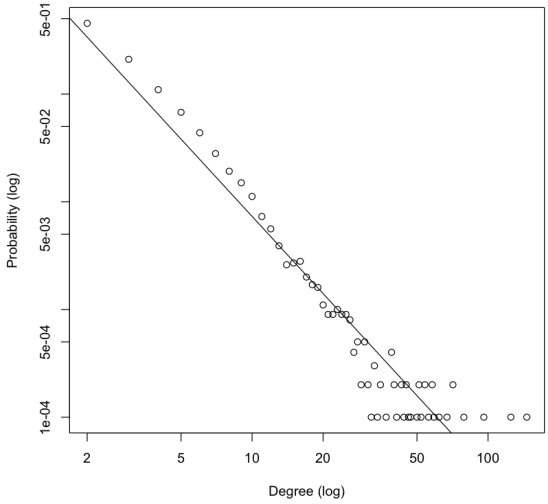
(b) Use fast greedy method to find the community structure. Measure modularity.
Solution:
The modularity is **0.523438262050059**

(c) Try to generate a larger network with 10000 nodes using the same model. Compute modularity. How is it compared to the smaller network's modularity?
Solution:
Modularity of larger network is **0.528133169317809**. As value of 'm' increases there isn't much affect in modularity with greater 'n', since both the number of nodes as well as greater connectivity increases, hence balancing out the effect of each other.

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

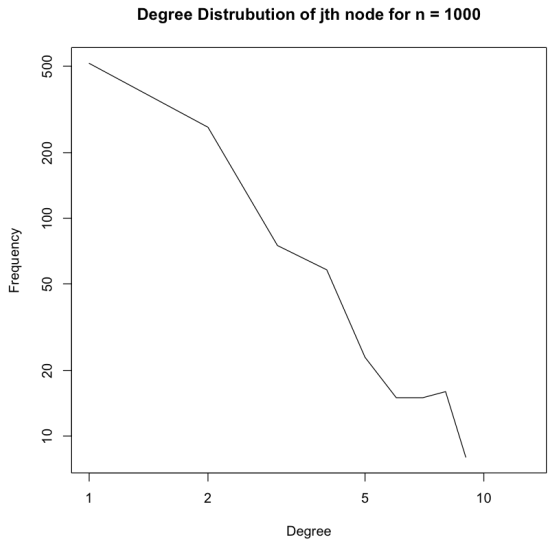
The slope of the plot is -2.0938
The slope of the plot for n = 10000 is -2.3813

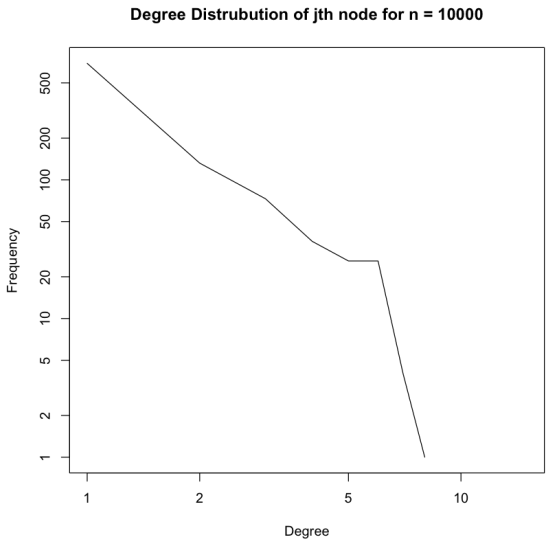




(e) You can randomly pick a node i, and then randomly pick a neighbor j of that node. Plot the degree distribution of nodes j that are picked with this process, in the log-log scale. How does this differ from the node degree distribution?

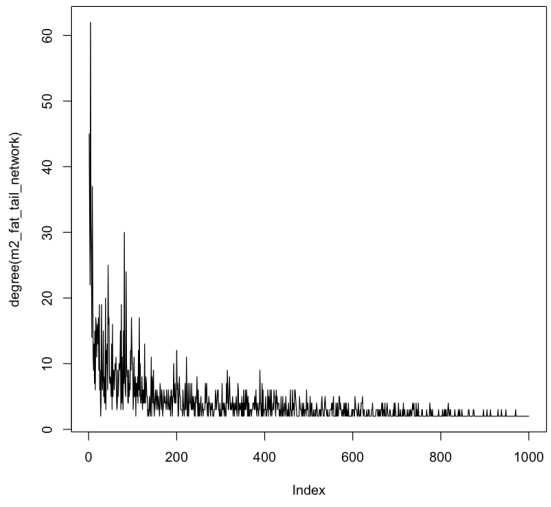
Solution:

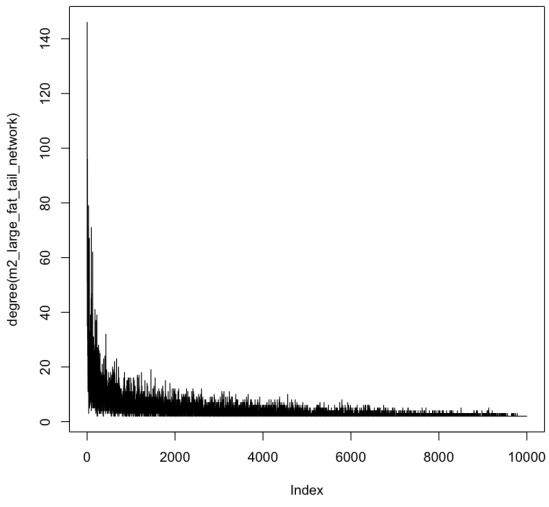




The node degree distribution exhibited power law perfectly, the above graph, although are in similar lines but doesn't fit with the power law distribution law well.

(f) Estimate the expected degree of a node that is added at time step i for 1 <= i <= 1000. Show the relationship between the age of nodes and expected degree through an appropriate plot.
Solution:





<u>For m=5:</u>

(a) Create an undirected network with n = 1000 nodes, with preferential attachment model, where each new node attaches to m = 5 old nodes. Is such a network always connected?

Solution:

Yes, it is always connected.

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

Solution:

The modularity is **0.280045995559396**

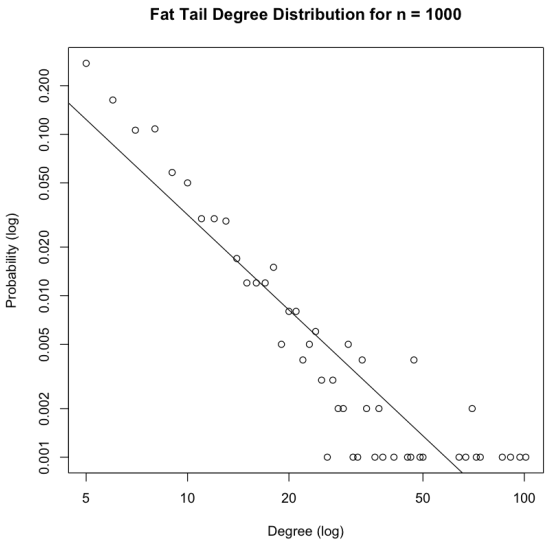
(c) Try to generate a larger network with 10000 nodes using the same model. Compute modularity. How is it compared to the smaller network's modularity?

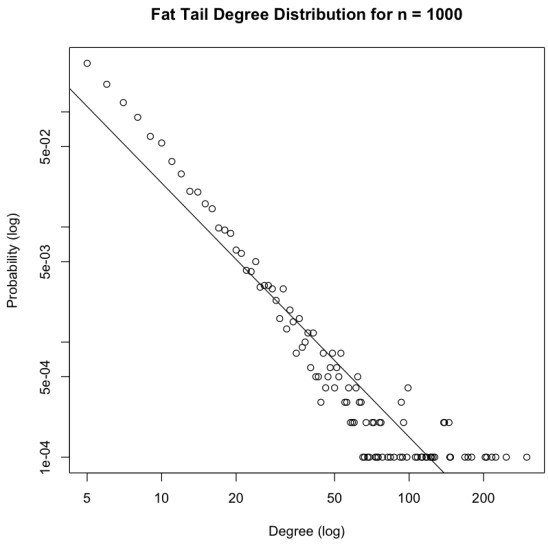
Solution:

Modularity of larger network is **0.271989880449182**. As value of 'm' increases there isn't much affect in modularity with greater 'n', since both the number of nodes as well as greater connectivity increases, hence balancing out the effect of each other.

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

Solution:



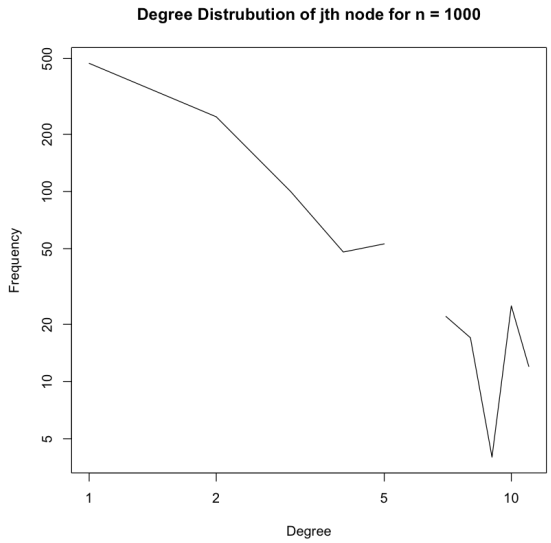


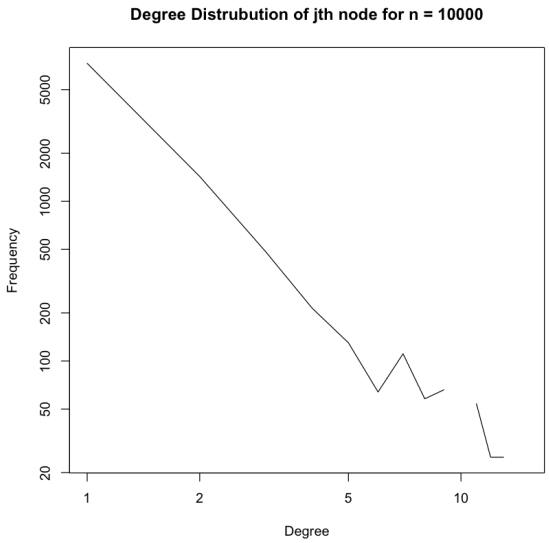
The slope of the plot is -1.9591

The slope of the plot for n=10000 is -2.2074

(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?

Solution:

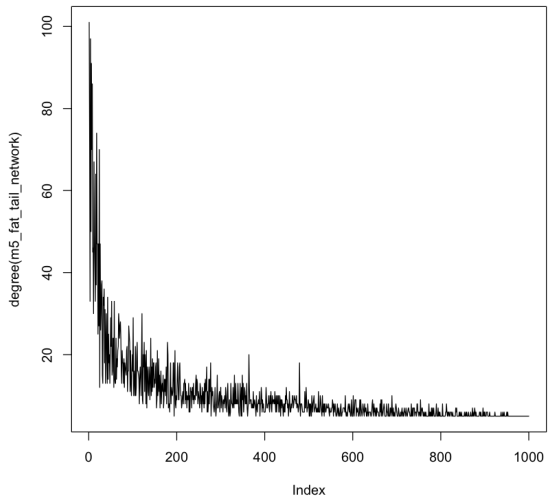


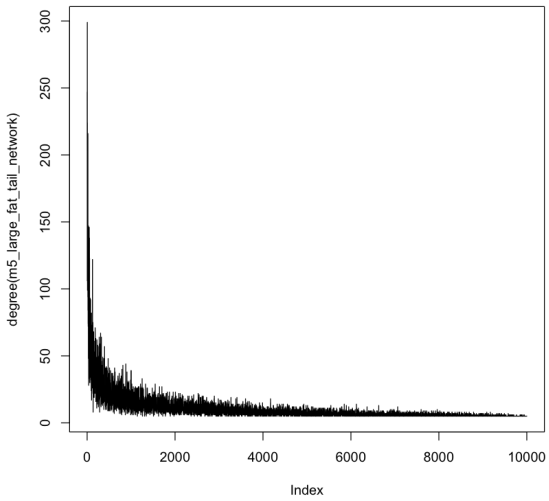


The node degree distribution exhibited power law perfectly, the above graph, although are in similar lines but doesn't fit with the power law distribution law well.

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

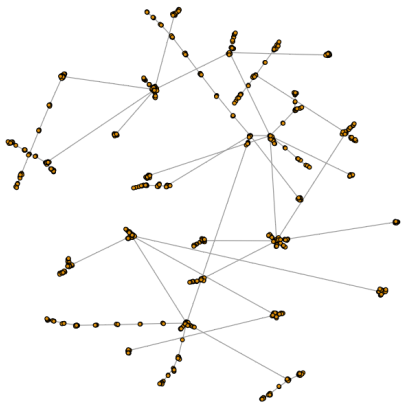
Solution:

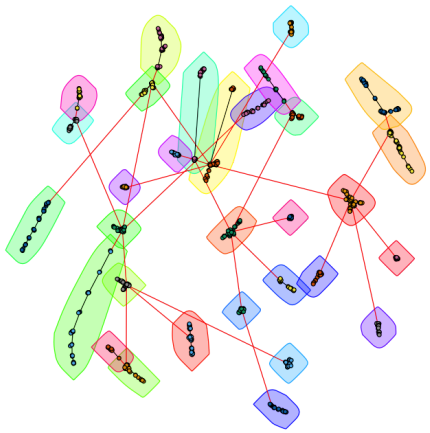


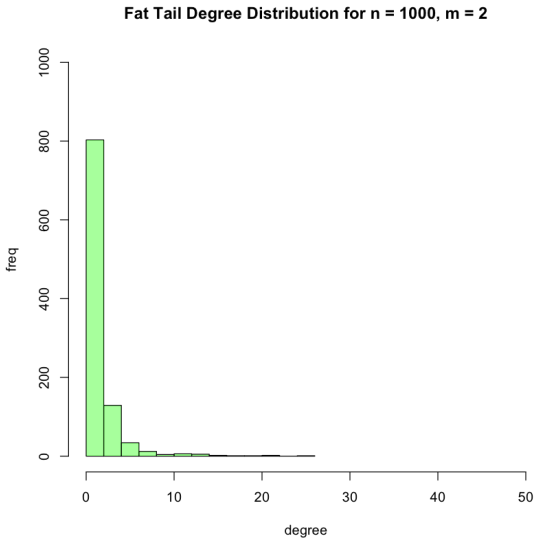


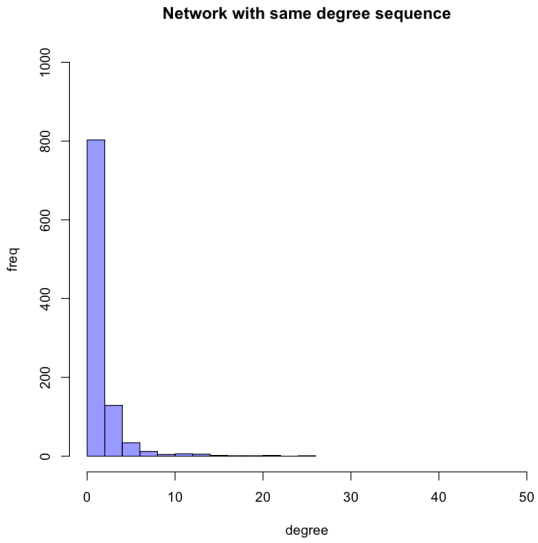
(h) Again, generate a preferential attachment network with n = 1000, m = 1. Take its degree sequence and create a new network with the same degree sequence, through stub-matching procedure. Plot both networks, mark communities on their plots, and measure their modularity. Compare the two procedures for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

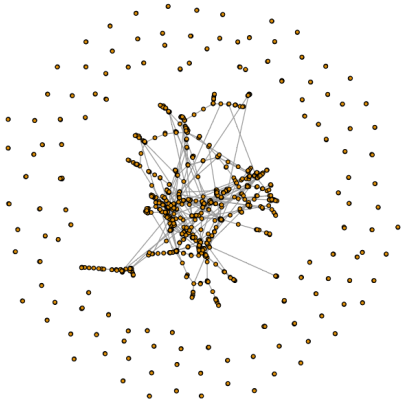
Solution:

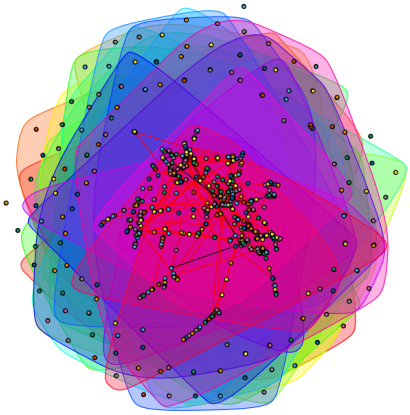












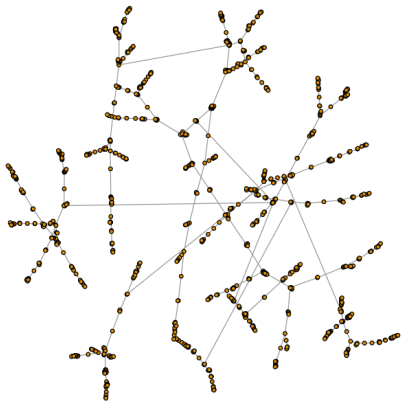
Modularity of first network is **0.933361790218648**
Modularity of second network is **0.795255162776252**

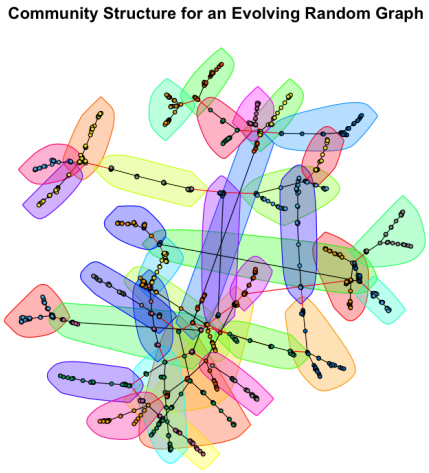
In the first network, we use preferential attachment model to create the network, which implicitly follows power law, whereas in second we impose power law by using the degree sequence of a model that creates power law distribution. Although degree sequence is same, we can see preferential model has higher modularity. The community structure graphs also depict the same. Hence, we can conclude that the degree sequence can only contribute in distribution but not in clustering or partitioning; two graphs with same degree sequence can be both highly clustered and loosely partitioned, depending on what procedure was used to create them.

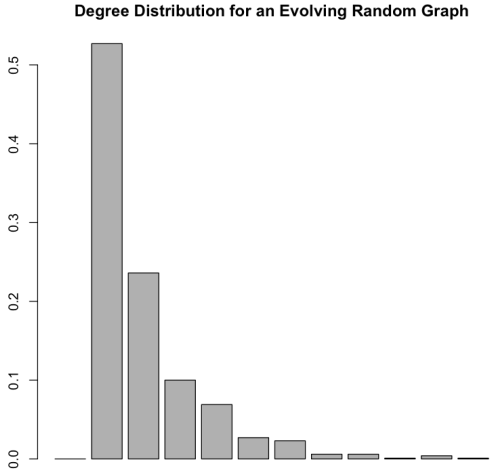
3. Create a modified preferential attachment model that penalizes the age of a node

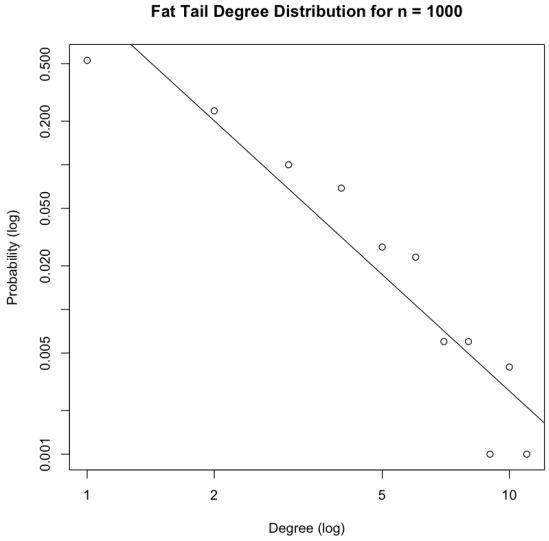
(a) Each time a new vertex is added, it creates m links to old vertices and the probability that an old vertex is cited depends on its degree (preferential attachment) and age. Produce such an undirected network with 1000 nodes. Plot the degree distribution. What is the power law exponent?

Solution:









The power law exponent is **-2.6706**

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

Solution:

Modularity is 0.935213491770052

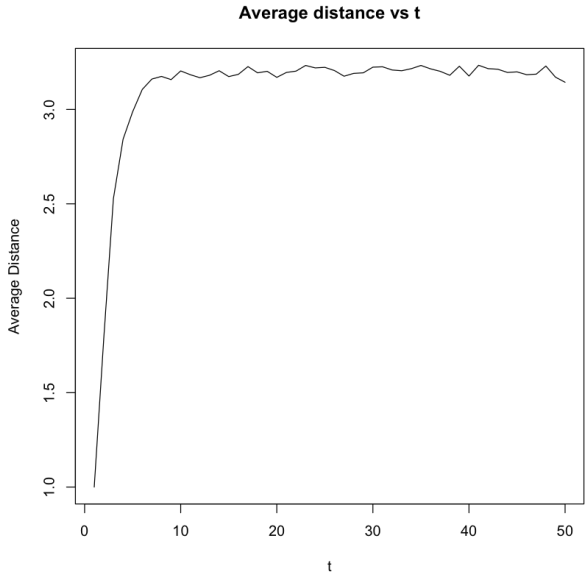
Part 2: Random Walk on Networks

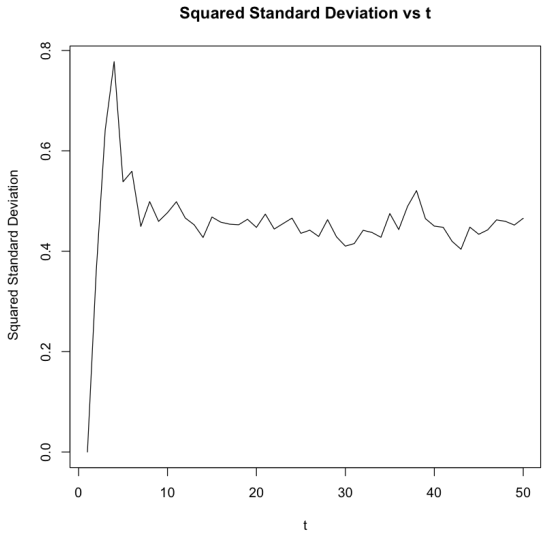
Q1. (a) Create an undirected random network with 1000 nodes, and the probability p for drawing an edge between any pair of nodes equal to 0.01.

Solution:
We create an undirected graph(using erdos.renyi.game function) with 1000 nodes, and p=0.01. We see that the graph is connected and has an edge count of 4980 and diameter of graph=5.

Q1. (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) $\langle s(t) \rangle$ of the walker from his starting point at step t. Also, measure the standard deviation $\sigma 2(t) = \langle (s(t) - \langle s(t) \rangle) 2 \rangle$ of this distance. Plot $\langle s(t) \rangle$ v.s. t and $\sigma 2(t)$ v.s. t. Here, the average $\langle \cdot \rangle$ is over random choices of the starting nodes.

Solution:
In this part, we conduct multiple(for 1000 randomly selected starting nodes) random walks for 50 steps each. The plots are as below:



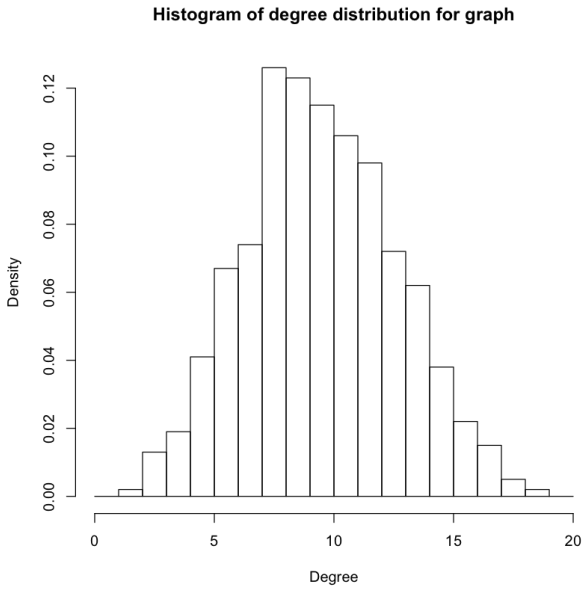


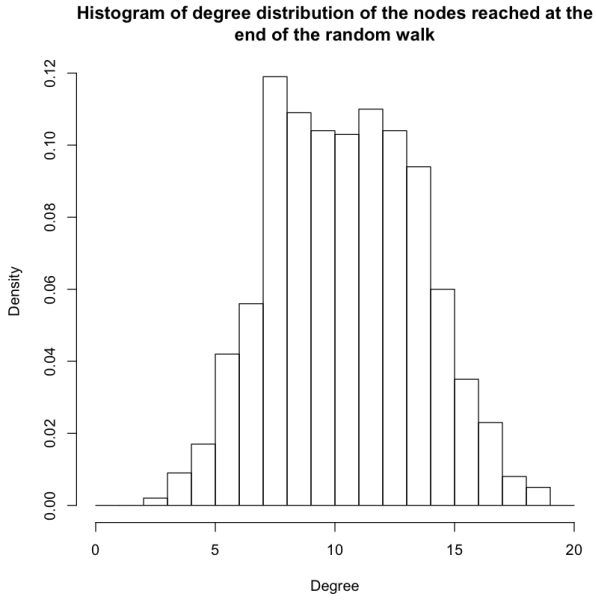
Observation: We observe from the plots that in this case, (n=1000, diameter=5) the squared standard deviation(stabilizes around 0.4) and average distance both converge fairly quickly with t, i.e. after certain value of t, not much variation is seen in either of them. This makes sense as since diameter is low (around 5), average distance (defined as the shortest path length, somewhere between 3 and 3.5 (around 3.2)in the plot above) will always be upper bounded by

the diameter upper bounded by the diameter, irrespective of the number of steps we take and converge soon.

Q1. (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?

Solution: In this part we plot histograms for degree distribution of graph as well as degree distribution of nodes reached at the end of random walk





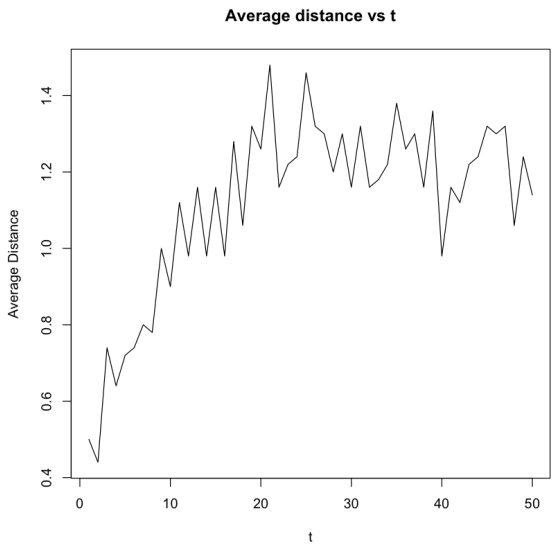
Observation: We observe the 2 histograms plotted seem very similar i.e the degree distribution of nodes reached at the end of random walk seems similar to that of the graph. This makes sense because (once one does the random walk enough times, we did 1000 times for n=1000), you would approximately have each node as the end node, and hence the distribution would look similar to that of the actual graph.

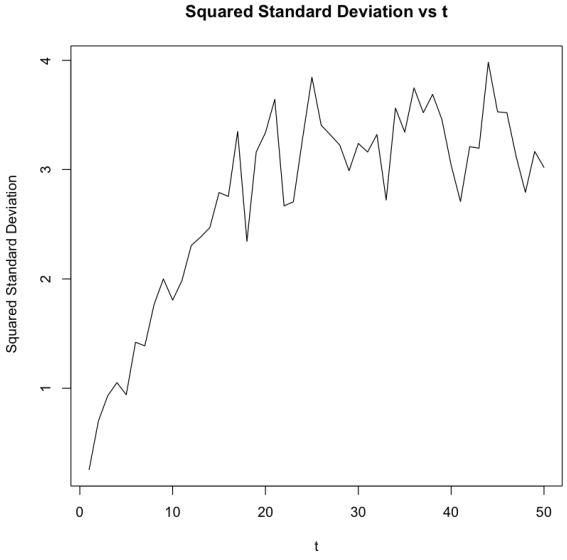
Q1. (d) Repeat (b) for undirected random networks with 100 and 10000 nodes. Compare the results and explain qualitatively. Does the diameter of the network play a role?

Solution:

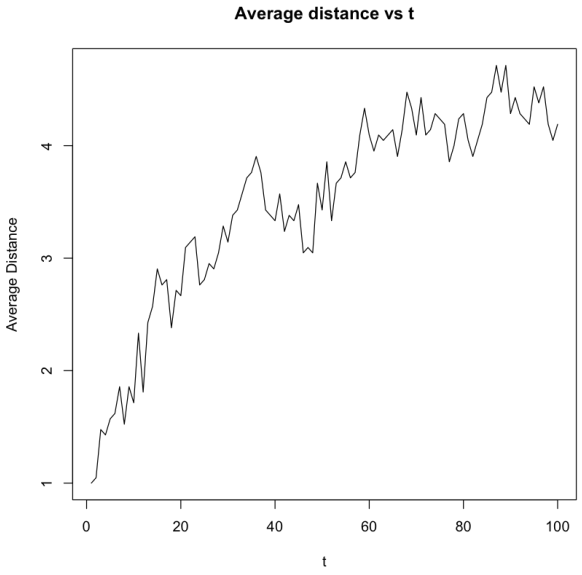
For n=100, we notice that the graph is unconnected. So we do 2 cases, one where we do the random walk over the unconnected network itself and second, when we do the random walk over the gcc of the original graph. The results are as follows:

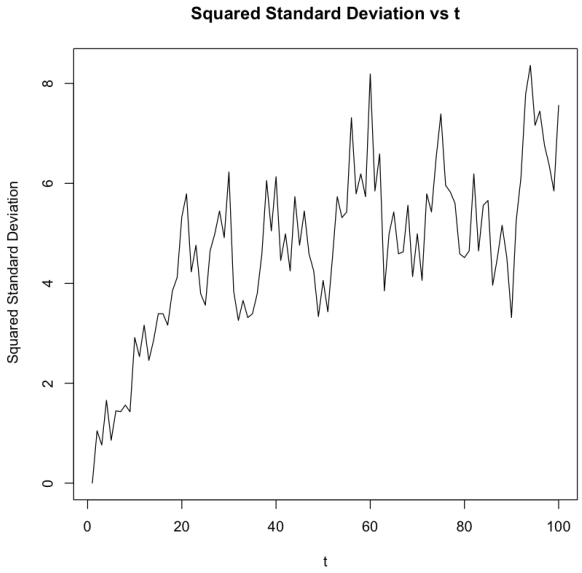
CASE 1:Random walk over the unconnected network itself:



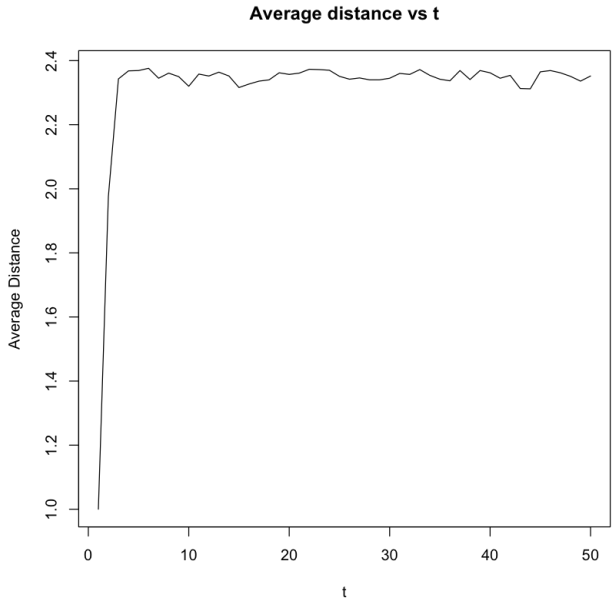


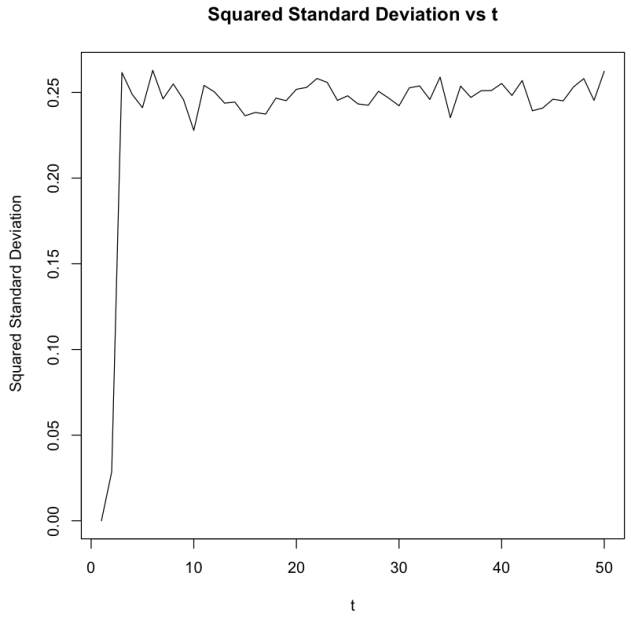
CASE 2:Random walk over the gcc of the original graph:





For n=10000, the results are as follows:





Observations:

First we notice that for n=100, graph is unconnected and mentioned before consider 2 cases. We notice that the average distance is greater in case 2 than case 1 which makes sense because in case 1, we may get stuck in some disconnected part leading to less average distance.

The relationship between diameter and number of nodes is as below
n=100, Diameter:10
n=1000, Diameter: 5
n=10000,Diameter :3

This makes sense that diameter decreases as number of nodes increase as the graph becomes well connected with increase in number of nodes is this case.

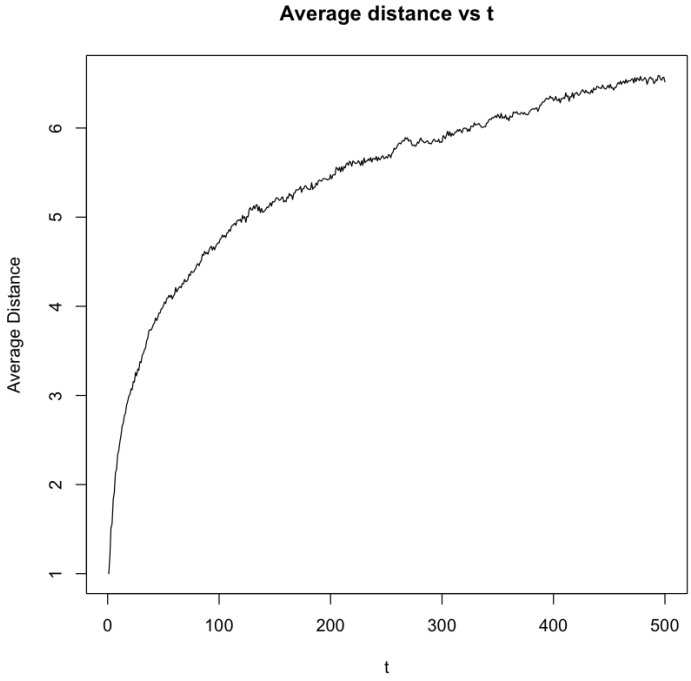
We also observe that when (n=10000, diameter=3) the squared standard deviation(stabilizes around 0.25) and average distance both converge very quickly with t, i.e. after certain value of t, not much variation is seen in either of them. This makes sense as since diameter is low, average distance (defined as the shortest path length, somewhere between 2.2 and 2.4(around 2.3) in the plot above) will always be upper bounded by it irrespective of the number of steps we take and converge very soon.
On the other hand, when n=100 and diameter is high, we observe that it takes more steps(we have to take till t=100 in this case to see convergence) for the values of average distance(stabilizes at somewhere between 4 and 5(around 4.5)) and squared standard deviation(between 4 and 8) to converge.
The n=1000 case lies in between these 2(explained previously in part b)

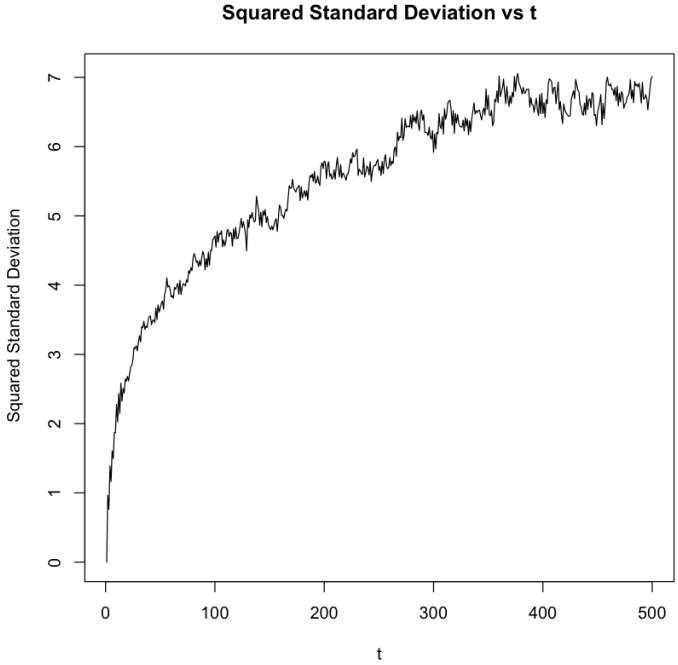
We can hence conclude that the smaller the diameter, the faster the average distance and standard deviation converge to value(stabilize).

Q2. (a) Generate an undirected preferential attachment network with 1000 nodes, where each new node attaches to m = 1 old nodes.
Solution:
We create an undirected graph(using barabasi.game function) with 1000 nodes, and p=0.01. We see that the graph is connected and has an edge count of 999 and diameter of graph=20.

Q2. (b) Let a random walker start from a randomly selected node. Measure and plot $\langle s(t) \rangle$ v.s. t and $\sigma 2(t)$ v.s. T.

Solution: In this part, we conduct multiple(for 1000 randomly selected starting nodes) random walks for 500 steps each(We tried less but plots did not seem to be converging so tried till 500). The plots are as below:



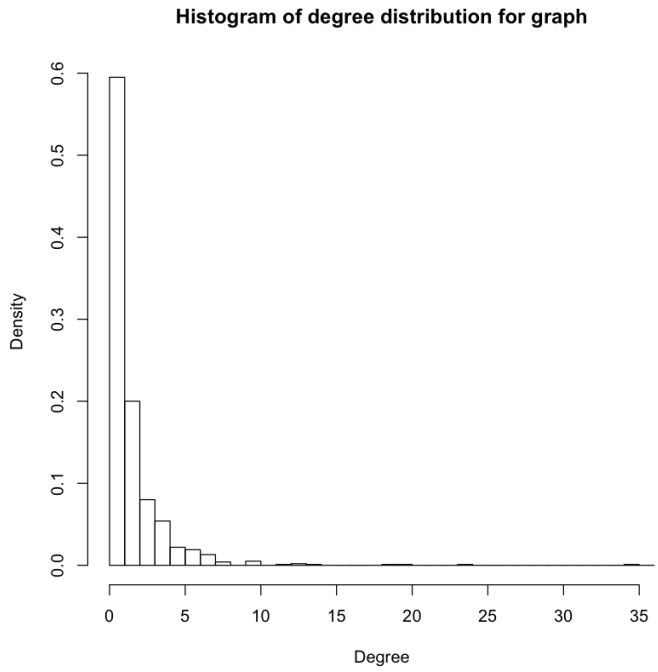


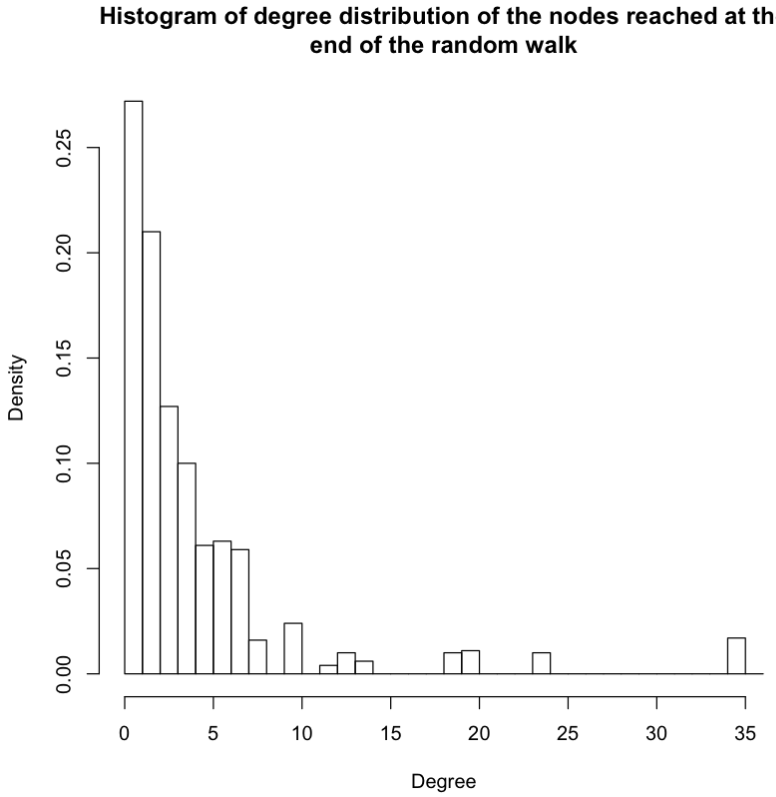
Observations:

We observe from the plots that in this case, (n=1000, diameter=20) the squared standard deviation(around 7 in the plot above) and average distance both take time to converge. (Average Distance does not seem to converge even till t=500) This makes sense as since diameter is high(around 20), average distance (defined as the shortest path length, somewhere between 6 and 7(around 6.5) in the plot above for 500 steps) , and squared standard deviation will also be high, and convergence will be slow. We can assume that the value of avg distance would stabilize eventually(as it should logically always be upper bounded by the diameterr).

Q2. (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?

Solution: In this part we plot histograms for degree distribution of graph as well as degree distribution of nodes reached at the end of random walk

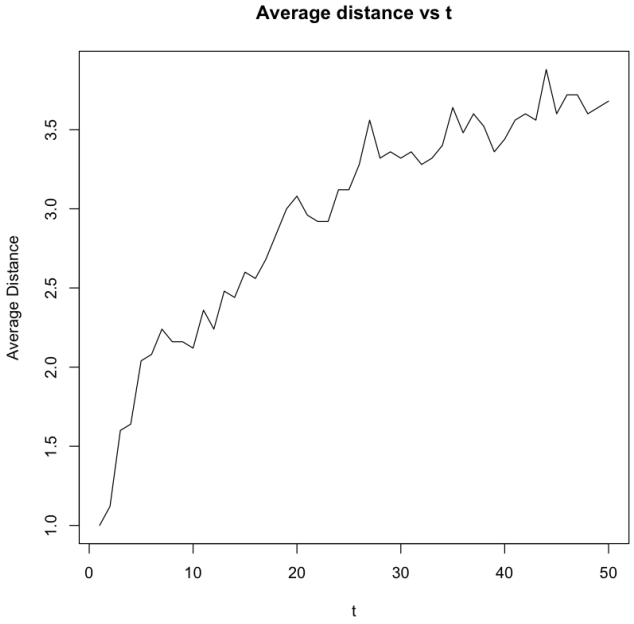


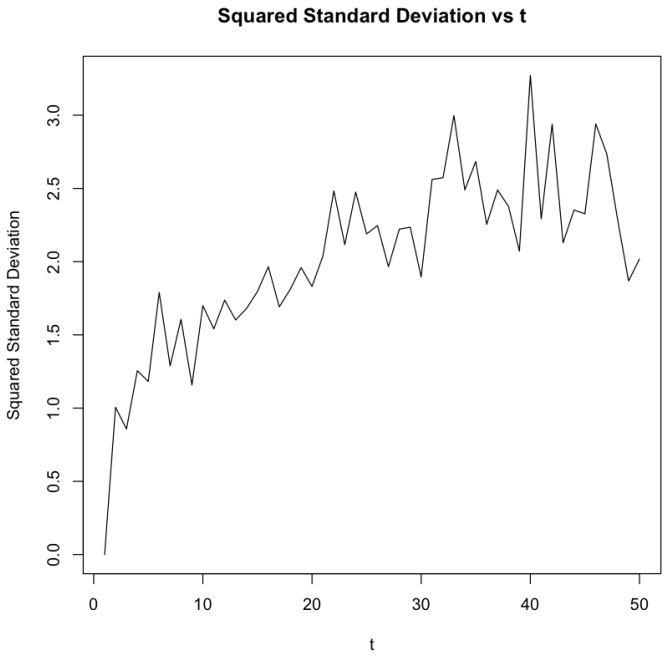


Observation: We observe the 2 histograms plotted seem very similar(in shape) i.e the degree distribution of nodes reached at the end of random walk seems similar to that of the graph. We also observe that nodes with higher degree occur relatively more when we plot histogram of degree distribution of nodes reached at the end of random walk. This make sense as nodes with higher degree are more likely to be the end node.

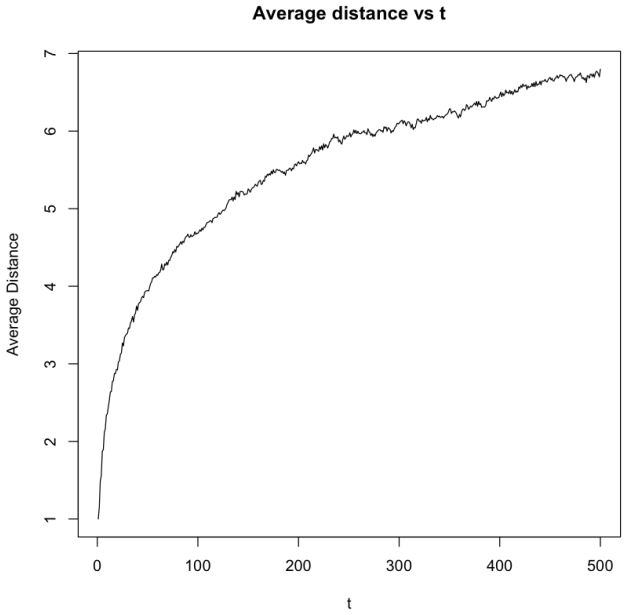
Q2. (d) Repeat (b) for preferential attachment networks with 100 and 10000 nodes, and m = 1. Compare the results and explain qualitatively. Does the diameter of the network play a role?
Solution:

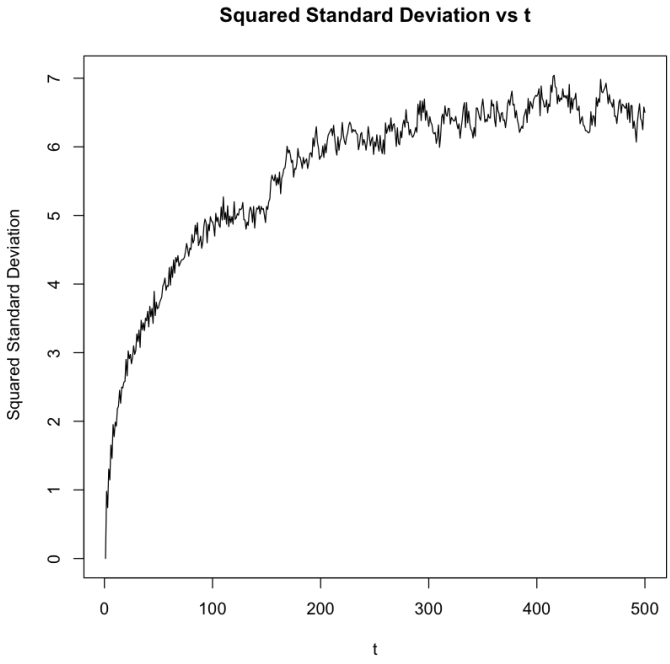
For n=100


Average distance vs t
Average Distance
1.0
1.5
2.0
2.5
3.0
3.5
0
10
20
30
40
50
t


Squared Standard Deviation vs t
Squared Standard Deviation
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
10
20
30
40
50
t

For n=10000





Observations:

The relationship between diameter and number of nodes is as below:
n=100, Diameter:11
n=1000, Diameter: 20
n=10000,Diameter :28

This makes sense that diameter increases as number of nodes increases as these are preferentially attached networks.

We observe that when (n=10000, diameter=28), the squared standard deviation(around 7 in the plot above) and average distance both take time to converge. (Average Distance does not seem to converge even till t=500) This makes sense as since diameter is high, average distance (defined as the shortest path length, somewhere between around 7 in the plot above for 500 steps), and squared standard deviation will also be high, and convergence will be slow. We can assume that the value of avg distance would stabilize eventually(as it should logically always be upper bounded by the diameter).
On the other hand, when n=100 and diameter is relatively low(diameter=11), we observe that it takes less steps(we have to take till t=100 in this case to see convergence as shown in fig 1 below) for the values of average distance(stabilizes at somewhere around 4) and squared standard deviation(around 4) to converge. While relatively less, diameter 11 is still high and can't be seen as converging properly with t=50 as shown above.

The n=1000 case lies in between these 2(explained previously in part b)

We can hence conclude that the smaller the diameter, the faster the average distance and standard deviation converge to value(stabilize).

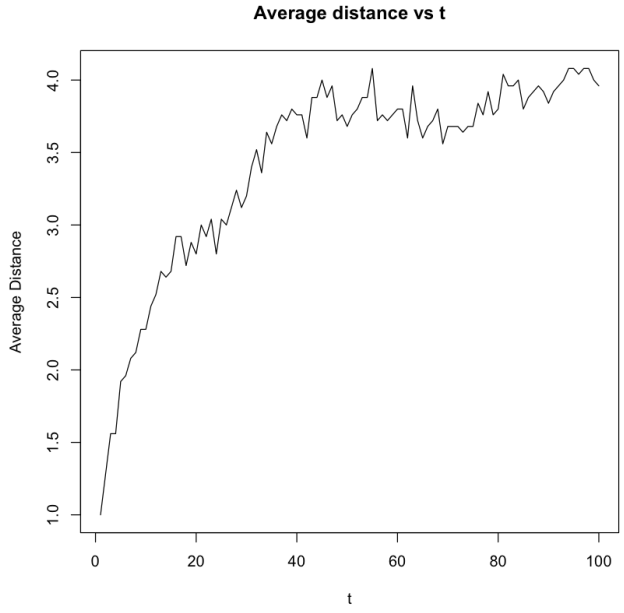


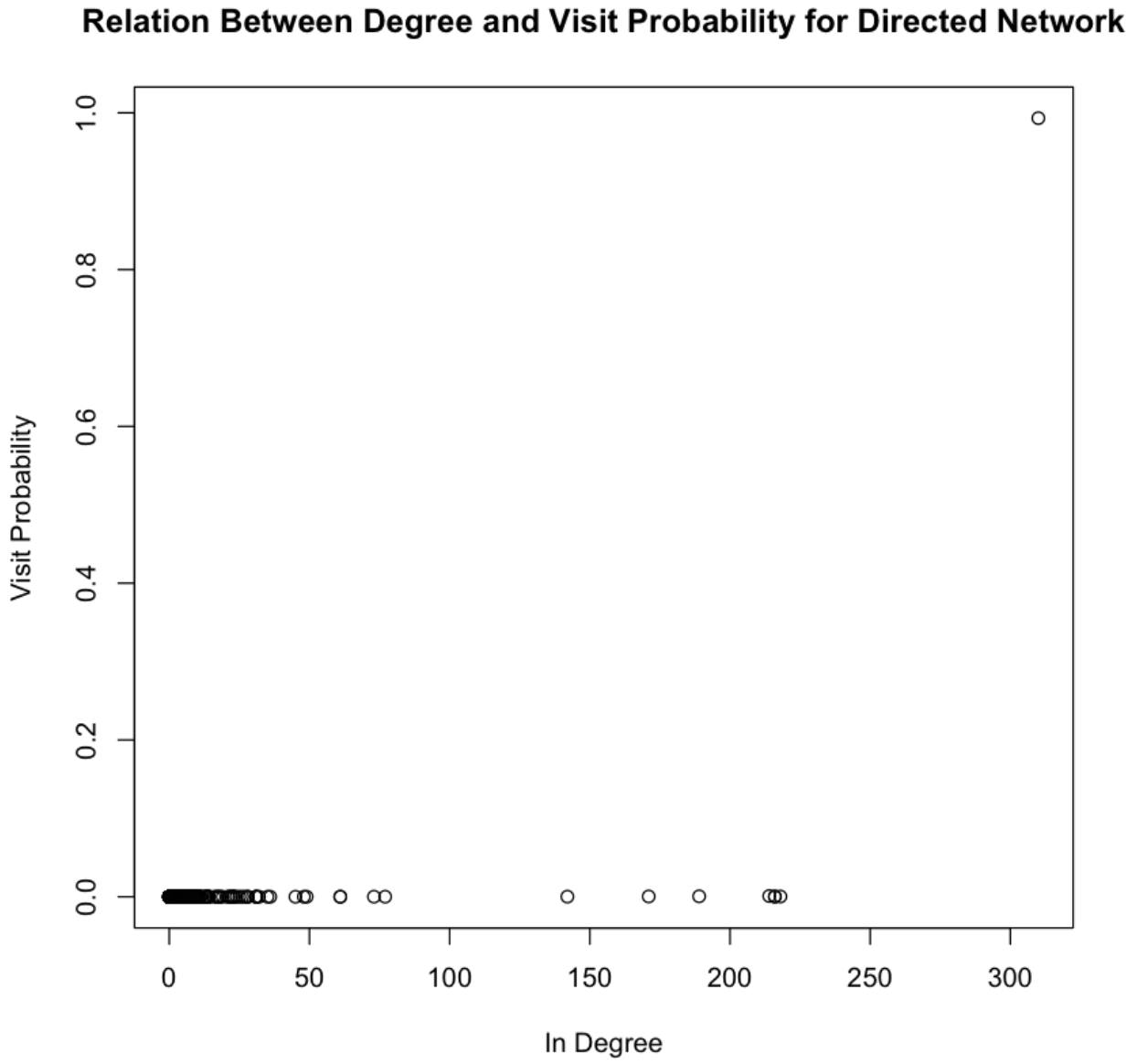
Fig 1

Q3. (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 created a directed network with 1000 nodes, using the preferential attachment model, where m=4. Based on the preferential network model, nodes with higher indegree have a higher probability to be connected to new incoming nodes. Hence a few heavily linked nodes tend to get heavier and heavier with time.

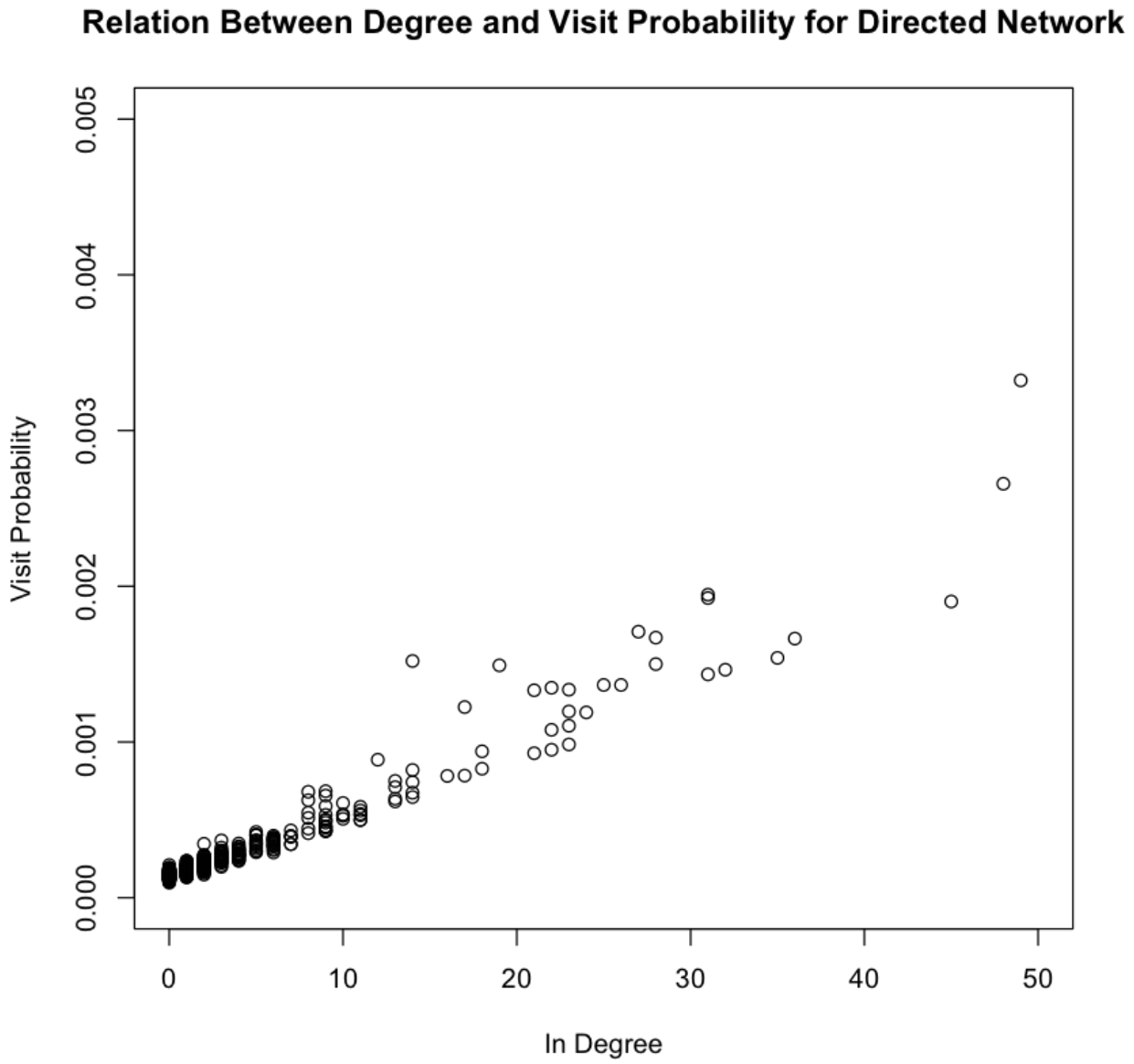
In order to measure the probability that the walker visits each node, random walk was run for 1000 iterations for 2000 time steps each. 2000 times steps were seen as sufficient as the **Yes, the probability is related to the indegree** because the Network seems to converge towards the ‘high-indegree-nodes’ and would rarely then move on to the unpopular (low-indegree) nodes. As such, following the observation that only a few number of nodes are the popular (high-indegree) nodes, and the majority of the nodes have a very less in-degree , the chances of visiting all of these unpopular nodes is expected to be negligent. This was replicated in our experiment as none of the 1000 iterations were able to cover all nodes. Hence based on both - theoretical expectations and practical experiments, we conclude the probability to be zero or negligent.



It was observed that the probability of visiting the first node was close to 1; while for the rest of the nodes it was close to 0. This is expected because the first node has the highest in-degree in the network and the outdegree is 0; implying once the random walker lands on the first node, it stays there.

Q3. (b) In all previous questions, we didn't 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?

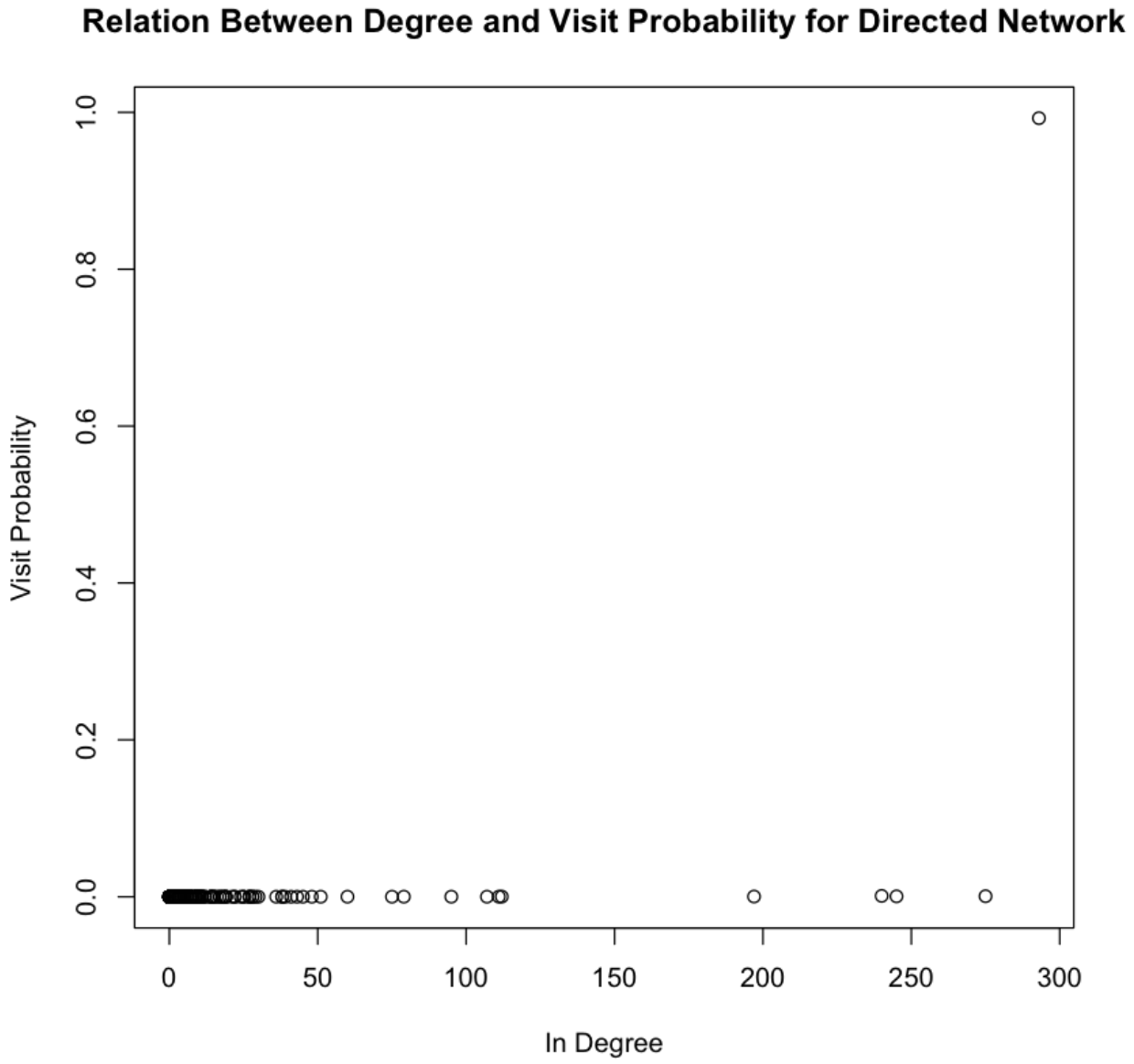
When we introduce teleportation, there is a 15% chance that the node won't use the transition matrix and instead randomly teleport to any node in the network. Since this random teleportation is not dependent on the indegree of the network , the probability to visit every node is now less dependent on the indegree but since there is only a 15% chance for this teleportation to happen, the dependence is still observed - only to a lesser degree than before.
For our network, the first node had just 60% probability of being visited and the rest of the probabilities were distributed across other nodes.



The same can be seen in the graph above. The nodes with smaller indegree have a nonzero probability of being visited over long timesteps.

Q4. (a) Suppose you have your own notion of importance. Your interest in a node is proportional to the node's 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).

When the teleportation probability to each node is proportional to the page rank, the chances of visiting the nodes with large page rank values is larger. Since the page rank vector is ~1 for first node and 0 for the rest of the nodes, this would mean the results will be similar to 3(a). Once again, when the random walker lands on the first node, it would teleport to itself only. This is reflected in the graph below as well:

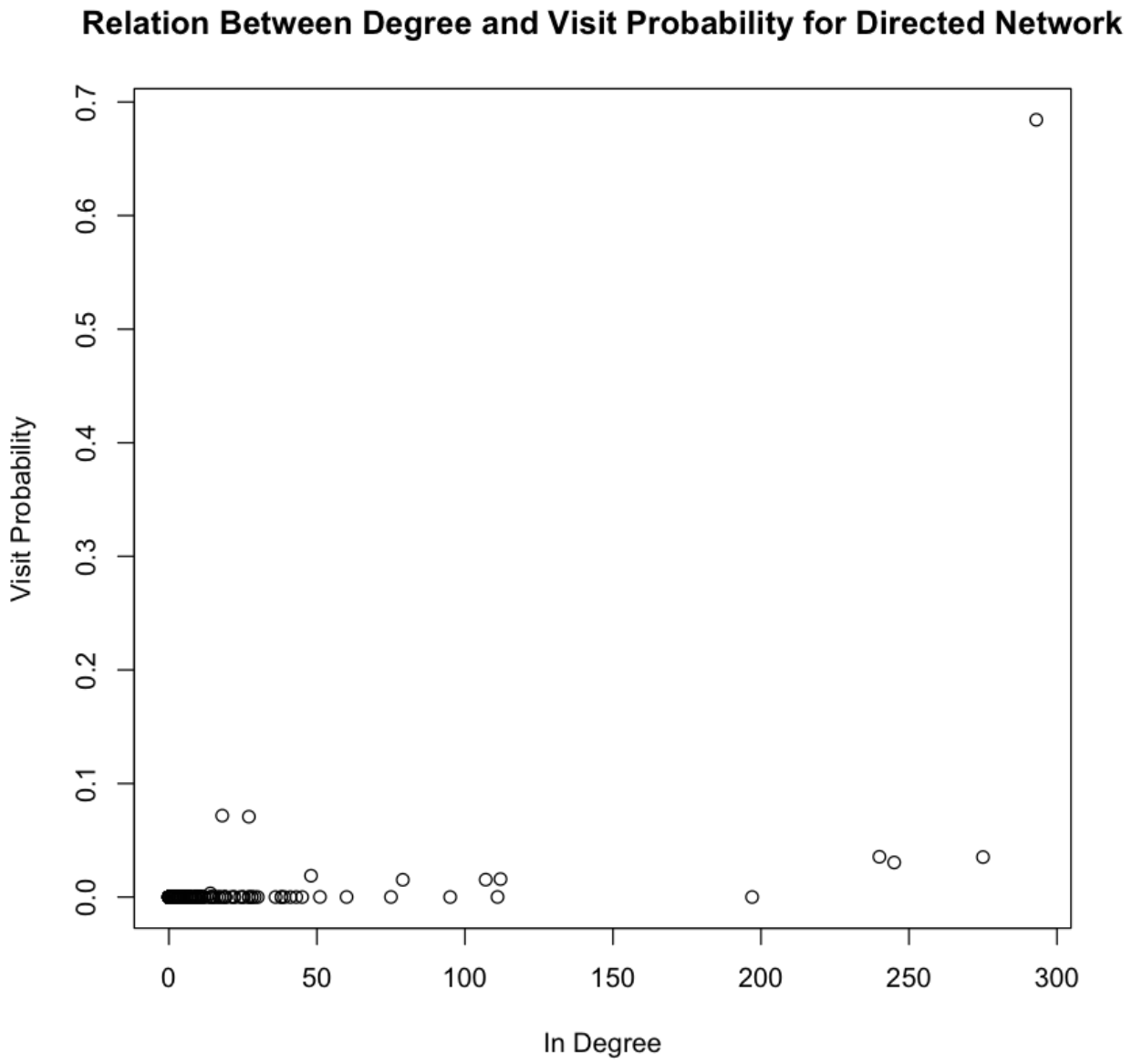


As expected, the probabilities tend to be biased towards the nodes with the highest indegree, and the probability that the entire network is visited is close to 0 for the same reason as 3a.

Q4. (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?

If teleportation landed only on two nodes with median PageRanks (probability ½ and ½) the PageRank values will change. The probability of visiting the first node is close to 70% and for the median nodes it is close to 7.5%. This is expected because α=0.15. So, majority of the

probability of visiting any node in the network is distributed between the first node and median nodes. This can be seen in the graph below where highest indegree node has highest probability followed by the two median nodes.



Q4. (c) More or less, (c) 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 people's interest in all nodes are the same. Can you take into account the effect of this self-reinforcement and adjust the PageRank equation?

$$PR(A) = \frac{(1-d)\ .I(A)}{T} + d\left(\frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \cdots\right).$$

d - decaying factor
I(A) - Indicator Function , I(A) = 1 if A is a trusted web page, else I(A) = 0
T - Total number of trusted web pages by the user

The first part of this equation represents the teleportation part. Assuming that the user trusts a total of T web pages, there is a 1/T probability for the user to land in one of the trusted web pages (Uniform probability assumed since no explicit distribution is mentioned in the question , else the term I(A)/T can be replaced with the probability distribution). To make sure that the probability is distributed only amidst the trusted web pages, an indicator function is multiplied to that part, so the teleportation only adds value to the page rank of the trusted web pages and not to the untrusted one as mentioned in the question. T could be a user specific heuristic that could be defined based in the indegree of nodes (more incoming edges means more trusted sites).

The second part is similar to the original page rank equation, as no changes by teleportation will occur in that part.