ECE 232E (Spring 2018)

Project 1: Random Graphs and Random Walks

Group Members:

Yunhao Ba (705032832)

Shuangyu Li (805035359)

Jingchi Ma (705027270)

Chenguang Yuan (005030313)

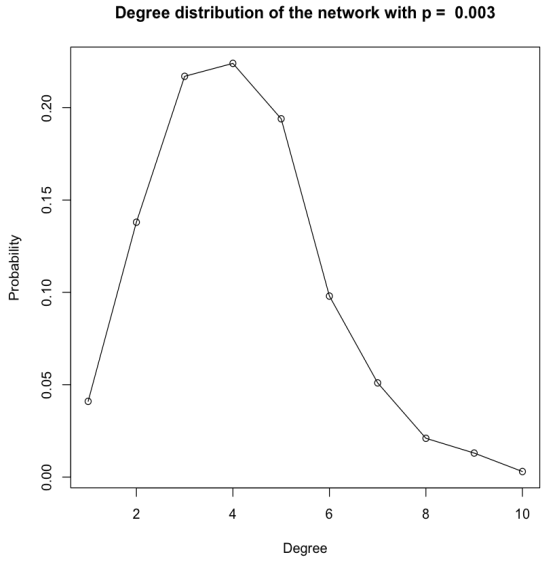
Table of Contents

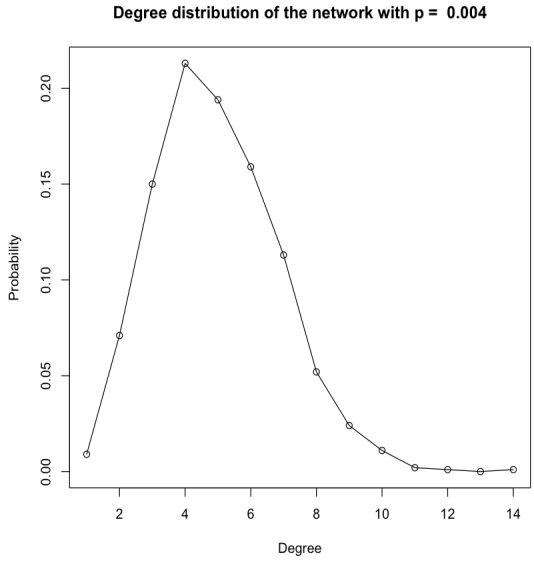
1. Generating Random Networks

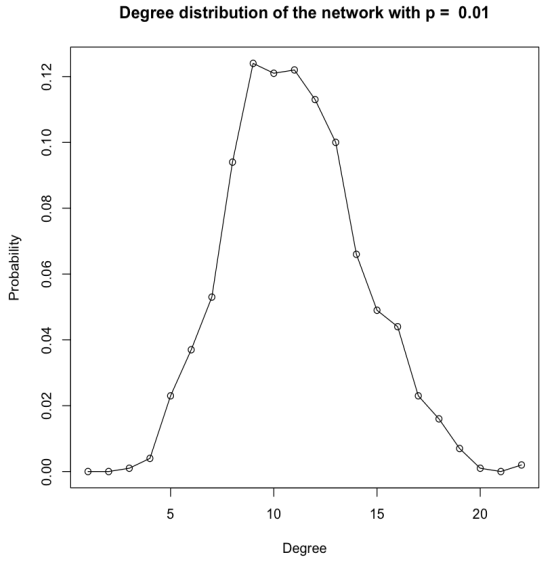
1.1 Create random networks using Erdös-Rényi (ER) model

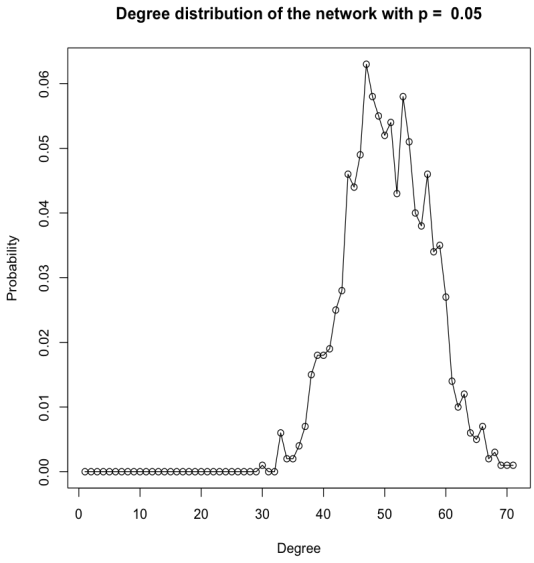
Part (a)

We create an undirected random networks with n = 1000 nodes, and the probability p for drawing an edge between two arbitrary vertices is 0.003, 0.004, 0.01, 0.05, and 0.1. We plot the degree distributions as follows:









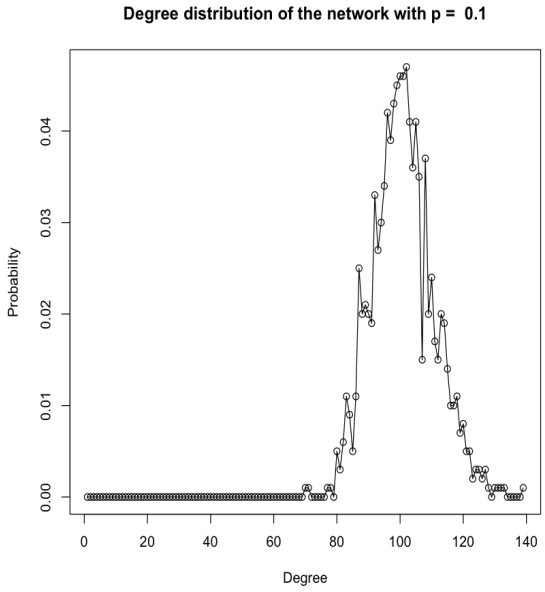


Figure: Degree Distribution of 5 Graphs with Probability p Taking Different Values

From the 5 plots above, we observed the trend of binomial distribution, since the distribution of the degree of any particular vertex is binomial.

The mean of binomial distribution is *np*; the variance of binomial distribution is *np(1-p)*.

Table: Comparison of Actual & Theoretical Means and Variances

p	Actual Mean	Theoretical Mean	Deviation	Actual Variance	Theoretical Variance	Deviation
0.003	2.886	3	3.8%	2.913917917	2.991	2.58%
0.004	4.13	4	3.25%	3.945045045	3.984	0.98%
0.01	9.834	10	1.66%	9.714158158	9.9	1.88%
0.05	50.038	50	0.076%	49.80235835	47.5	4.85%
0.1	100.592	100	0.592%	93.42496096	90	3.81%

We observed that there is only slight difference between the theoretical values and actual values. The mean and variance are close to each other in these cases, because the *p* values are relatively small.

Part (b)

Not all the random realizations of the ER network are connected.
Below we show the connectivity probability and the diameter of GCC for each *p*:

Table: The Connectivity Probability and Diameter of GCC

p	Connectivity Probability	Diameter of GCC (for one instance)
0.003	0	15
0.004	0	11

0.01	0.957	6
0.05	1	3
0.1	1	3

We can see that, as *p* gets larger, the connectivity probability gets larger, and diameter of GCC gets smaller.

Part (c)

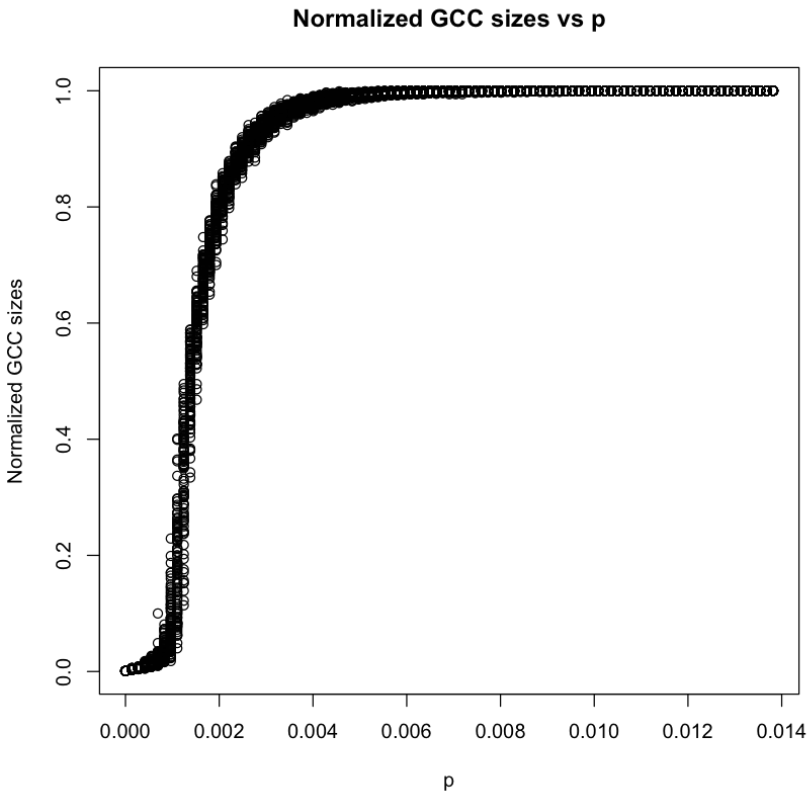


Figure: Normalized GCC Size versus *p*

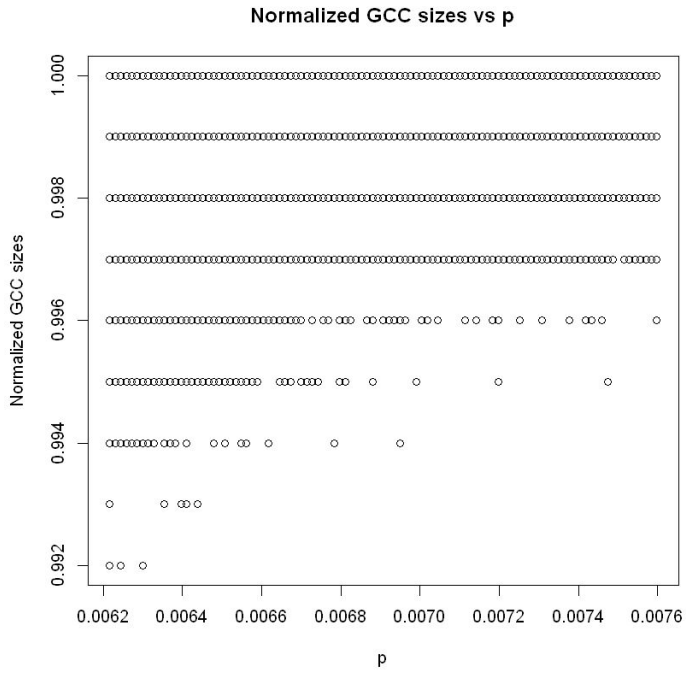


Figure: Normalized GCC Size versus *p (zoom in)*

We can define the emergence of GCC as follows:

In a graph G, if there exists a connected component C in the community, such that the sizes of all other connected components are *O(ln(n))*, where *n* is the vertex size of the graph, then we call the component C the Giant Connected Component(GCC) of graph G, and define the emergence of GCC accordingly.

Based on the figure above, we can see when p = 0.0065 the normalized GCC size ≈ 0.993, which means the size of other connected components is *O(ln(n))*. According to the definition above, we define the emergence of GCC at p = 0.0065.

Besides, we would like to check the probability that the network is connected to define the emergence of GCC. Therefore, we derived the plot below:

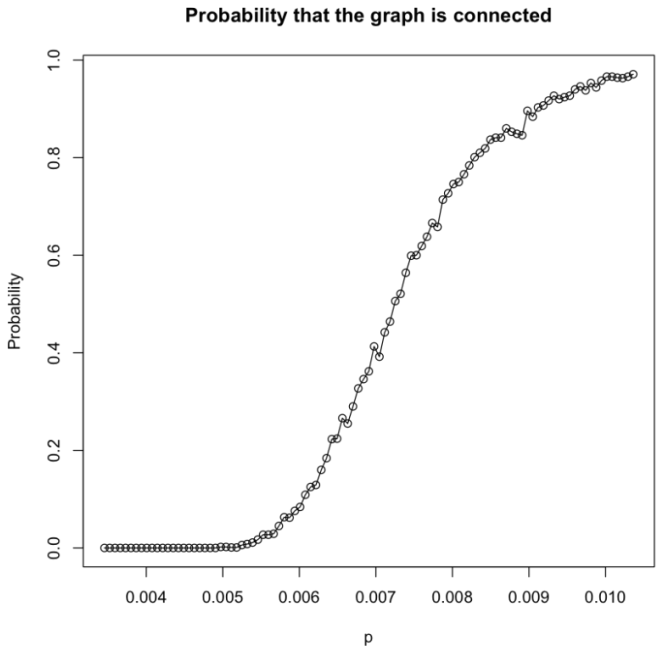


Figure: Probability that Network is Connected versus *p*

Alternatively, we can also define the inflation point of the above curve to be the point where a giant connected component starts to emerge.

We derived the inflection point to be 0.007529453, which is our estimation of the value of p where a giant connected component starts to emerge.

In the lecture, the professor shows that we can analytically derive the value of threshold probability (p_c) using the Erdos-Renyi Asymptotic Expression, the random graph Gp(N) is disconnected if the link density p is below the connectivity threshold using the formula:

$$p_c \propto \frac{ln(N)}{N} = \frac{ln(1000)}{1000} = 0.00691$$

As a result, our estimate value matches with the theoretical value, with a deviation of 8%.

Part (d)

(i)

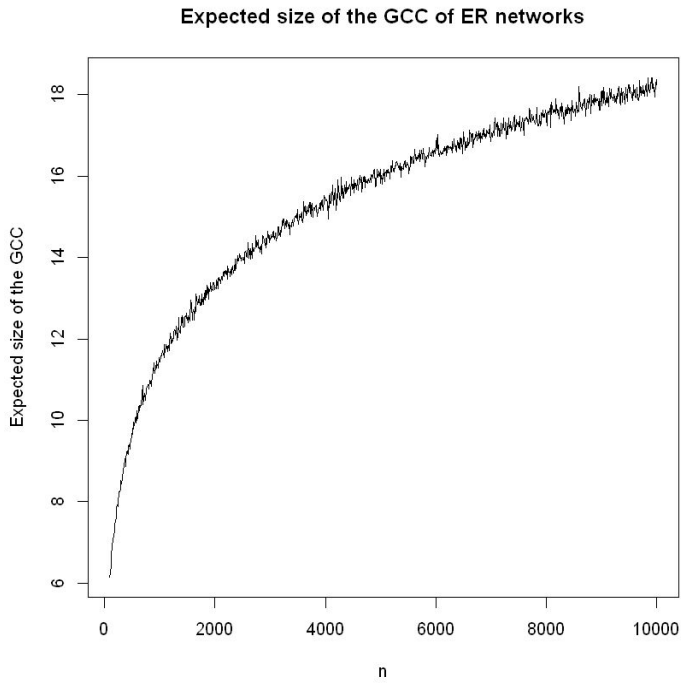


Figure: Expected Size of GCC versus *n* in ER Networks *(p=0.5/n)*

We can see from the plot above, in ER networks, as *n* gets larger, the expected size of GCC gets larger. The expected size of GCC is in the shape of O(ln(n)).

(ii)

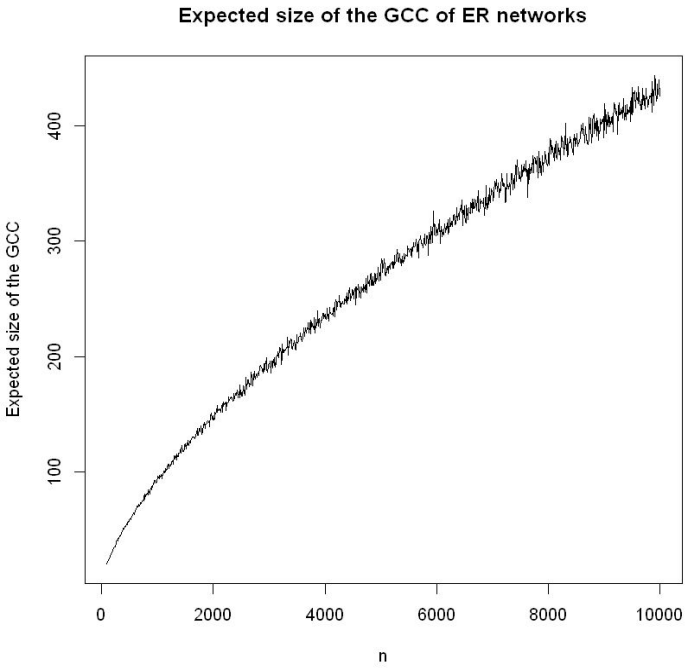


Figure: Expected Size of GCC versus *n* in ER Networks (*p=1/n*)

Compared to the case where c=0.5, this plot (c=1) has shown more linearity compared with c = 0.5. The expected size of GCC is in the shape of O($\sqrt{n}$).

(iii)

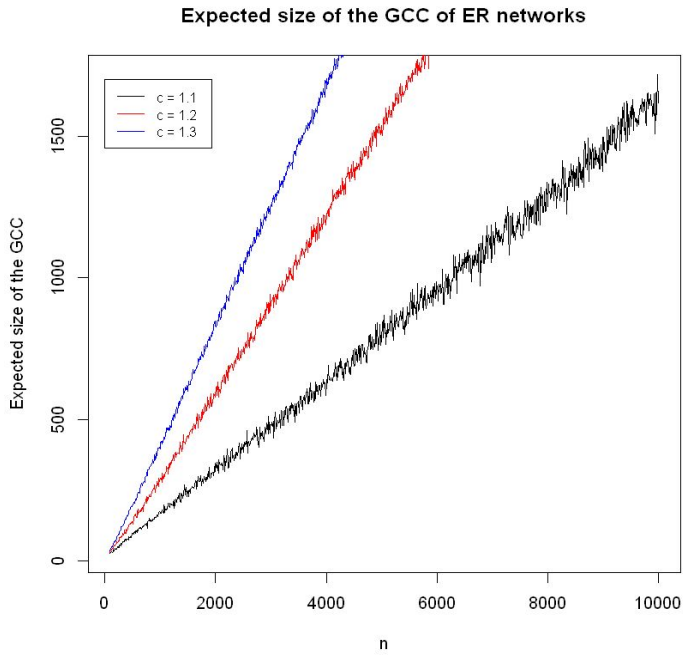


Figure: Expected Size of GCC versus *n* in ER Networks *(p=1.1/n, 1.2/n, 1.3/n)*

We can see that the plots have shown very good linearity. The expected size of GCC is linear in *n* when *c* is greater than 1. Also, as *c* gets larger, the slope gets larger.

1.2 Create networks using preferential attachment model

Part (a)

We created an undirected network with n = 1000 nodes, with preferential attachment model, where each new node attaches to *m = 1* old nodes. As a result, such a network is always connected.

Part (b)

We used fast greedy method to find the community structure:

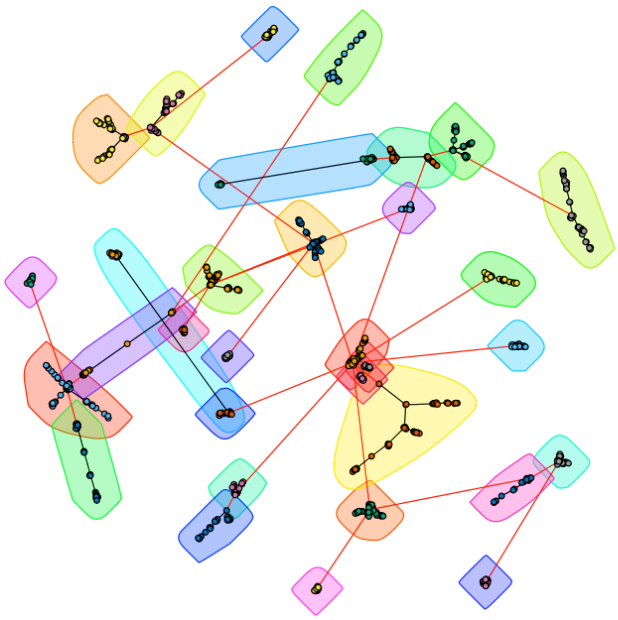


Figure: Community Structures in Preferential Attachment Network (n=1000, m=1)

Community sizes:
[56 55 58 42 47 43 39 40 44 43 36 35 33 31 30 32 28 27 26 25 24 23 27 21 19 21 27 18 18 16 16 14 13]

The modularity = 0.932389847304764

Part (c)

We then generated a larger network with 10000 nodes using the same model:

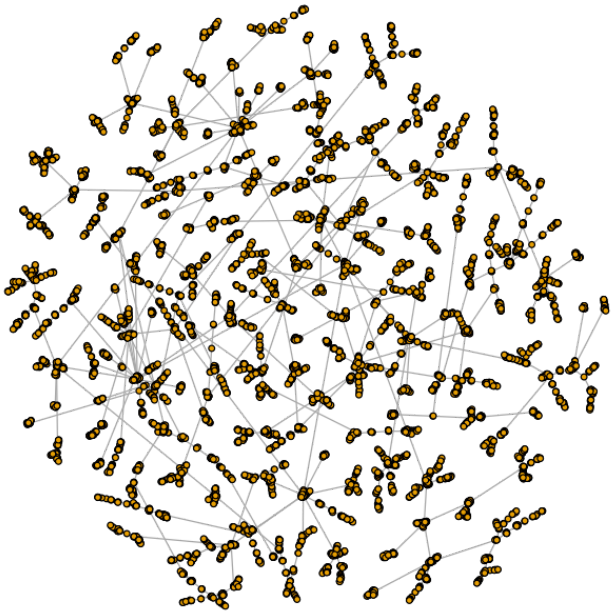


Figure: Network Generated Using Preferential Attachment Model (n=10000, m=1)

We used fast greedy method to find the community structure:

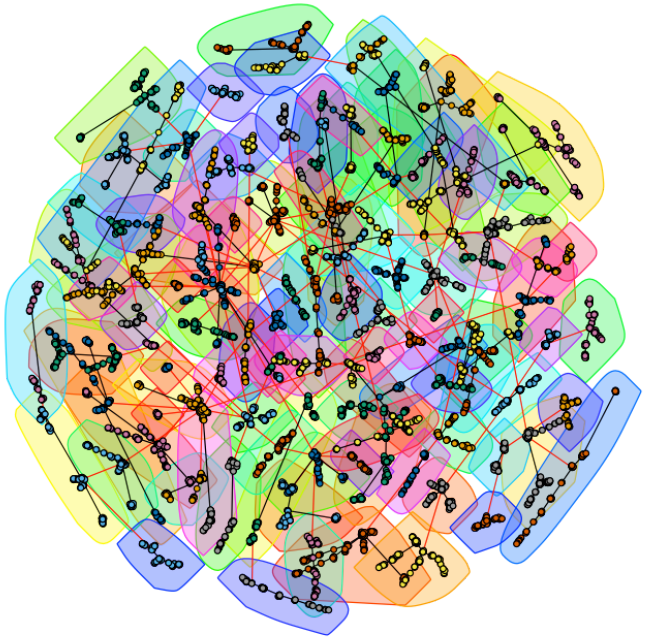


Figure: Community Structures in Preferential Attachment Network (n=10000, m=1)

Community sizes
[163 138 139 140 146 172 177 139 160 152 143 134 134 146 137 132 189 132 129 138 182 166 124 156 124 120 123 117 140 117 112 113 119 121 125 134 111 111 110 101 100 117 100 97 119 96 94 132 93 89 97 101 94 93 86 83 88 84 84 87 76 78 77 88 74 75 74 73 74 82 73 69 73 70 67 83 65 61 60 63 60 58 57 64 55 54 58 53 53 62 50 52 49 50 53 46 50 42 44 36 38 33 28]

The modularity = 0.97855750671578, which is larger compared to the smaller network's modularity.

Part (d)

n=1000:

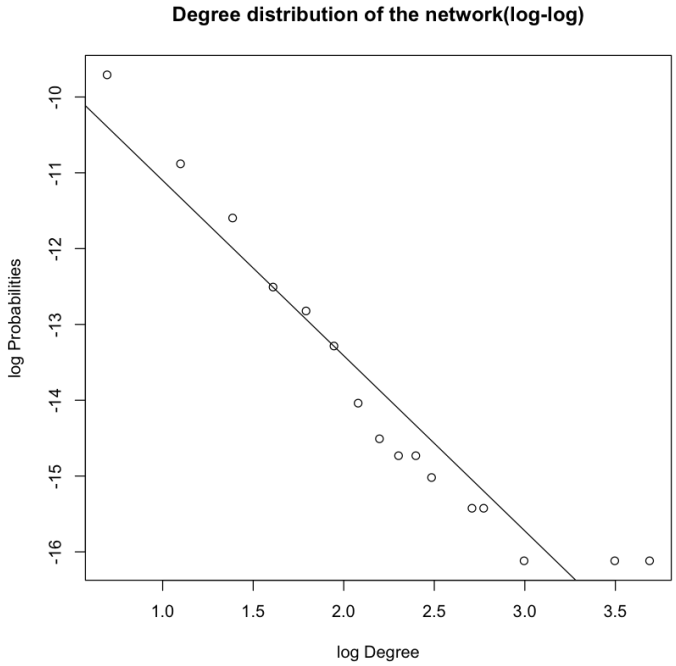


Figure: Degree Distribution in a Log-Log Scale (n=1000, m=1)

By linear regression, we have:

$$log\ (Probability) = -8.789 - 2.312\ log(Degree)$$

The estimated slope is -2.312

n=10000:

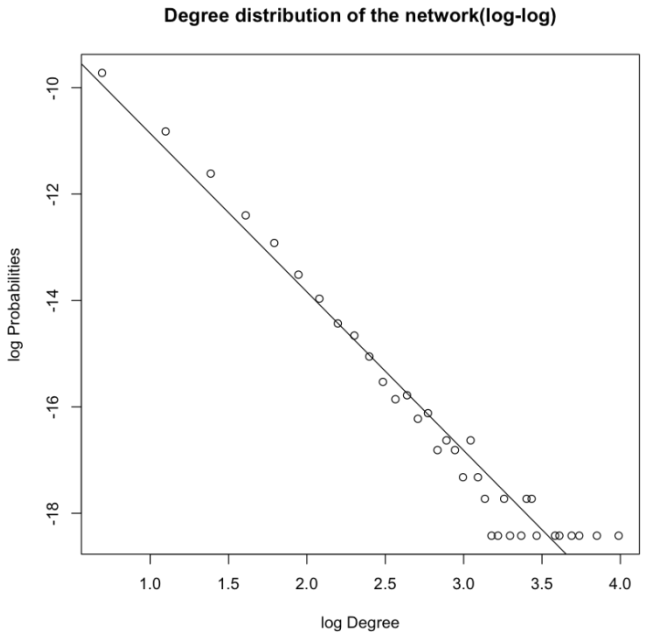


Figure: Degree Distribution in a Log-Log Scale (n=10000, m=1)

By linear regression, we have:

$$log\ (Probability)\ =-7.879-2.980\ log(Degree)$$

The estimated slope is -2.980

This result follows the theory that $P \propto \frac{C}{K^3}$. The result of 1000 nodes is a little bit off the theoretical result because of the deficient of the node. Because the equation of P is derived under the assumption of $k \to \infty$, p approaches to theoretical value under higher k.

Part (e)

For the 10000 nodes graph, we randomly pick a node i, and then randomly pick a neighbor j of that node. We plot the degree distribution of nodes j that are picked with this process, in the log-log scale:

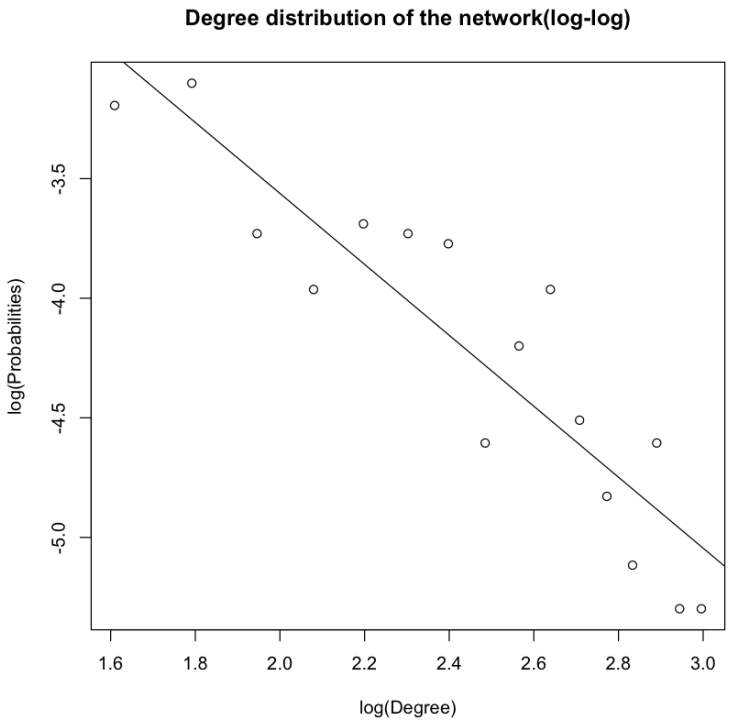


Figure: Degree Distribution of Nodes j in a Log-Log Scale (n=10000, m=1)

By linear regression, we have:

$$log\ (Probability)\ =-0.5953-1.4832\ log(Degree)$$

The estimated slope is 1.4832.

Compared to the results of part(d), the absolute value of slope of this plot is much smaller. Based on the theory, the probability to select point based on neighbor could increase the probability proportion to k. Because the nodes with higher degree has higher probability been selected,

$$P \propto \frac{C}{K^3} * k = \frac{C}{K^2}$$

Our power low result is a little bit smaller than the theoretical value may caused by the deficient of nodes which result in lots of zero frequency in high degrees.

Part (f)

We want to estimate the expected degree of a node that is added at time step i for $1 \leq \square \leq 1000$. The result follows both theory and intuition that the elder nodes are more likely to has higher degree.

The relationship between the age of nodes and their expected degree is shown below:

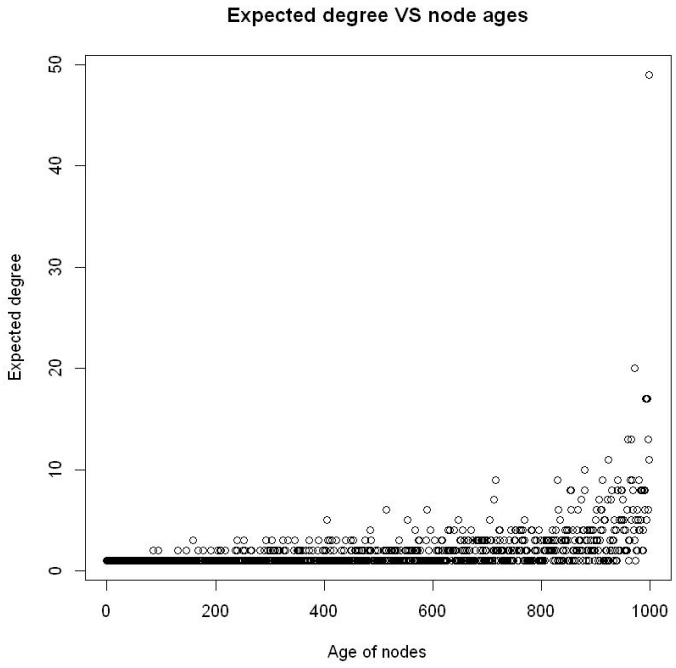


Figure: Expected Degree vs Age of Nodes (m=1)

Part (g)

m = 2:

We created an undirected network with n = 1000 nodes, with preferential attachment model, where each new node attaches to $m = 2$ old nodes. As a result, such a network is always connected.

We used fast greedy method to find the community structure:

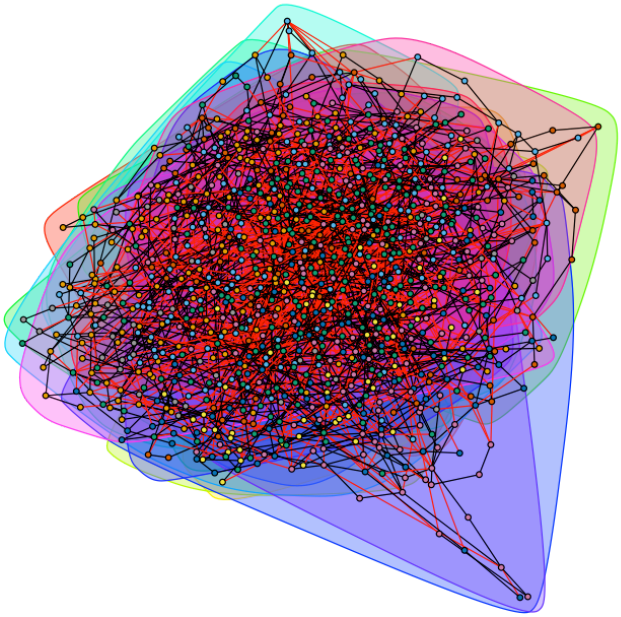


Figure: Community Structures in Preferential Attachment Network (n=1000, m=2)

The community sizes:
[63 30 29 31 28 40 86 31 38 67 96 84 49 56 62 63 115]

The modularity = 0.522030040554068. It is lower than the modularity of m = 1. Modularity measures the division of the subgraph. Based on the calculation equation of modularity shown below

$$Q=1/(2m) * sum((Aij-ki*kj/(2m)) delta(ci,cj),i,j)$$

m is the the edges number of the graph. A is adjacency matrix. k is degree. c is the component of a node. delta is i == j. With more edges it is naturally to have smaller modularity. It also follows intuition. When a new node tries to attach to more nodes, it may become a connection between different communities and thereby reduce the modularity of the whole graph.

We then generated a larger network with 10000 nodes using the same model：

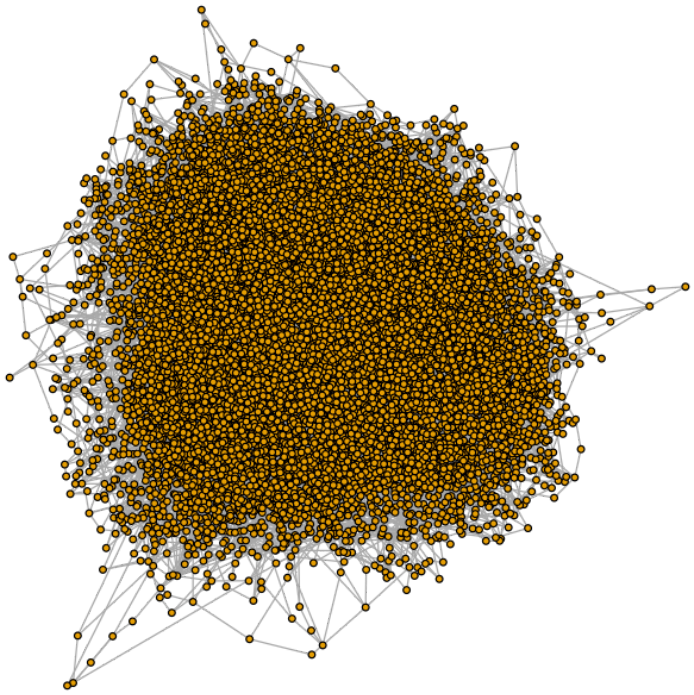


Figure: Network Generated Using Preferential Attachment Model (n=10000, m=2)

We used fast greedy method to find the community structure:

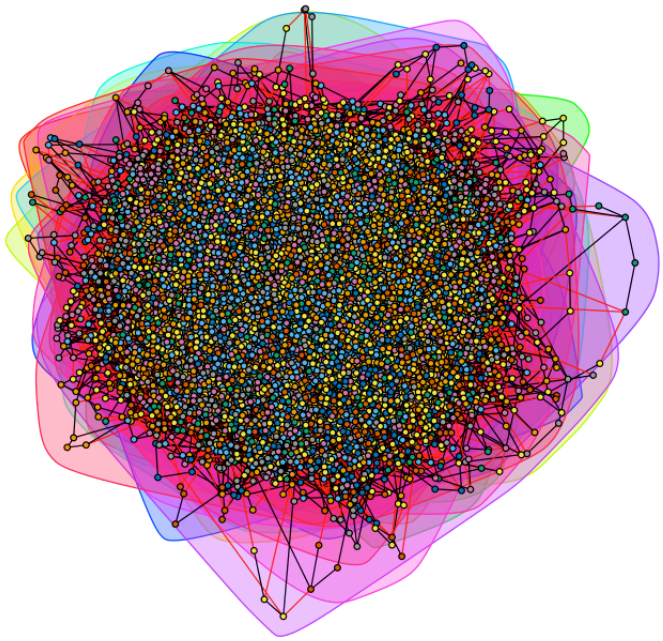


Figure: Community Structures in Preferential Attachment Network (n=10000, m=2)

Community sizes:
[123 107 121 261 89 189 278 177 85 251 150 173 119 127 164 109 828 95 48 729 140 174 161 305 183 275 313 607 407 404 578 684 637 909]

The modularity = 0.530928787940493, which is larger compared to the smaller network's modularity.

We plotted the degree distribution in a log-log scale for both n = 1000, 10000; then estimated the slope of the plots:

n = 1000:

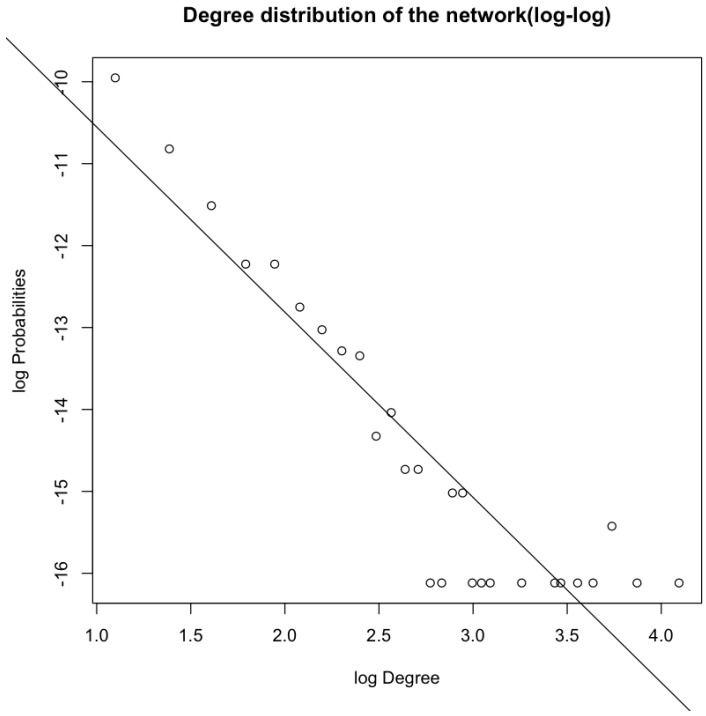


Figure: Degree Distribution in a Log-Log Scale (n=1000, m=2)

By linear regression, we have:

$$log\,(Probability) = -8.292 - 2.261\,log(Degree)$$

The estimated slope is -2.261

n=10000:

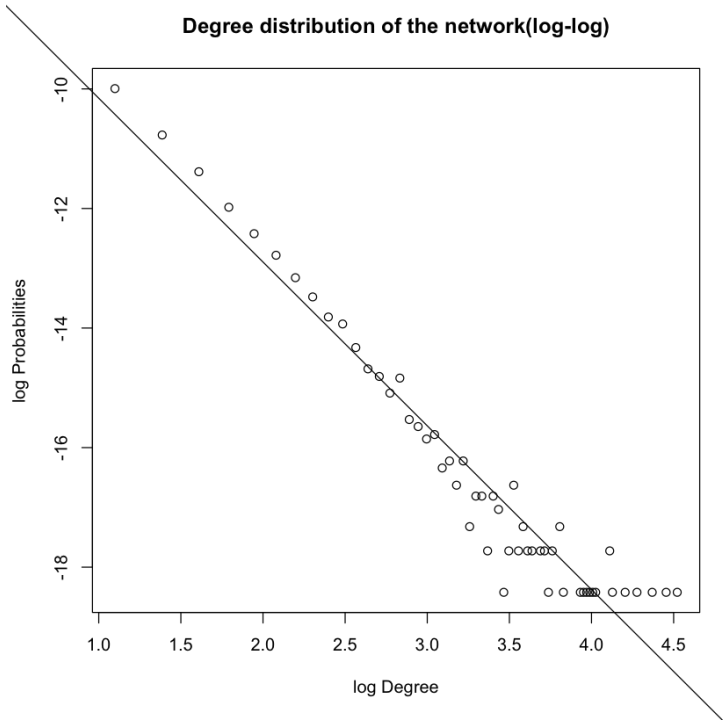


Figure: Degree Distribution in a Log-Log Scale (n=10000, m=2)

By linear regression, we have:

$$log\ (Probability) = -7.415 - 2.740\ log(Degree)$$

The estimated slope is -2.740

Next, we randomly pick a node i, and then randomly pick a neighbor j of that node. We plot the degree distribution of nodes j that are picked with this process, in the log-log scale:

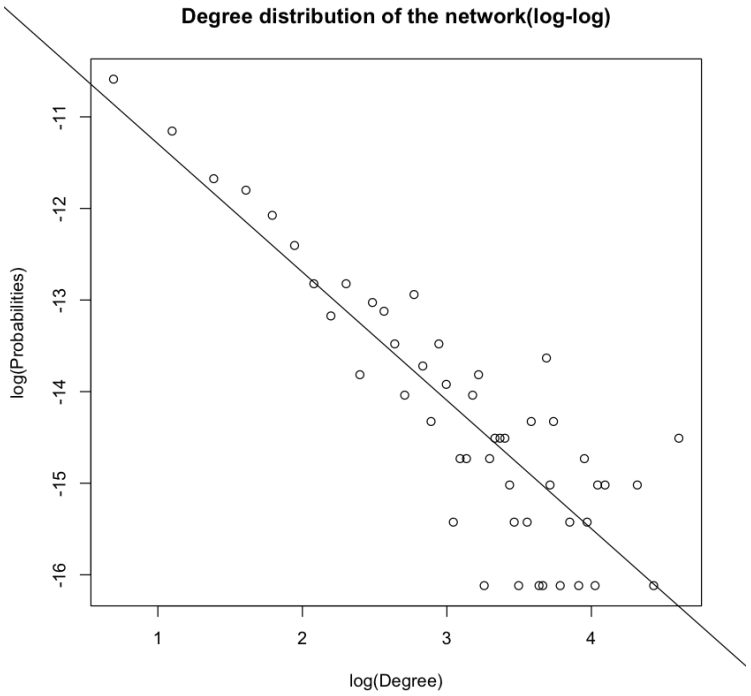


Figure: Degree Distribution of Nodes j in a Log-Log Scale (n=10000, m=2)

By linear regression, we have:

$$log\ (Probability) = -9.892 - 1.401\ log(Degree)$$

The estimated slope is **-1.401**

Compared to the results of the previous part, the absolute value of slope of this plot is much smaller.

We then want to estimate the expected degree of a node that is added at time step i for $1 \leq \square \leq 1000$. The relationship between the age of nodes and their expected degree is shown below:

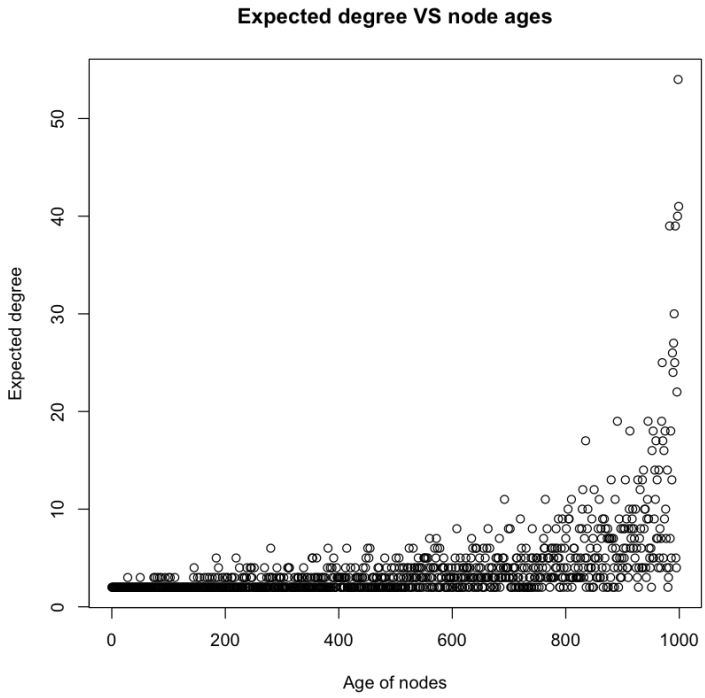


Figure: Expected Degree vs Age of Nodes (m=2)

m = 5:

We created an undirected network with n = 1000 nodes, with preferential attachment model, where each new node attaches to $m = 5$ old nodes. As a result, such a network is always connected.

We used fast greedy method to find the community structure:

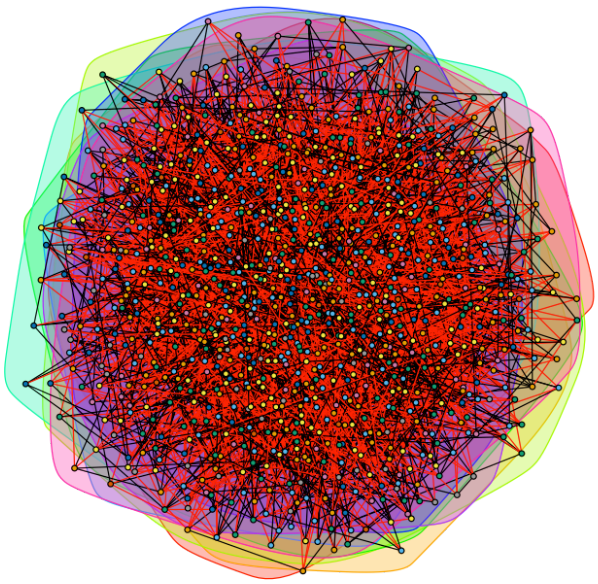


Figure: Community Structures in Preferential Attachment Network (n=1000, m=5)

Community sizes:
[50 188 160 185 119 17 65 55 161]

The modularity = 0.275498753029399.

We then generated a larger network with 10000 nodes using the same model：

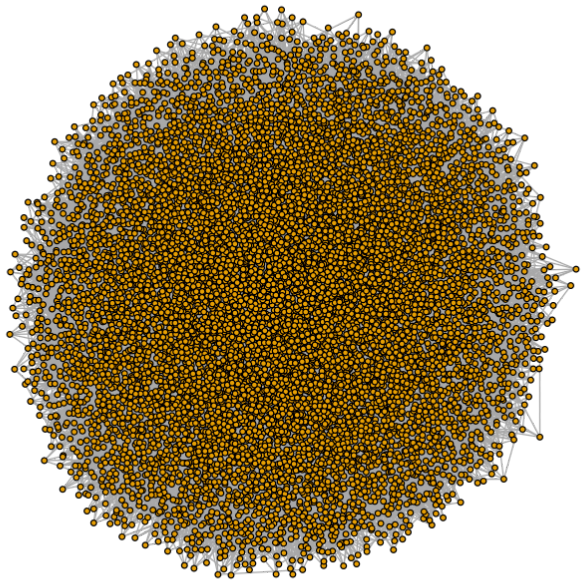


Figure: Network Generated Using Preferential Attachment Model (n=10000, m=5)

We used fast greedy method to find the community structure:

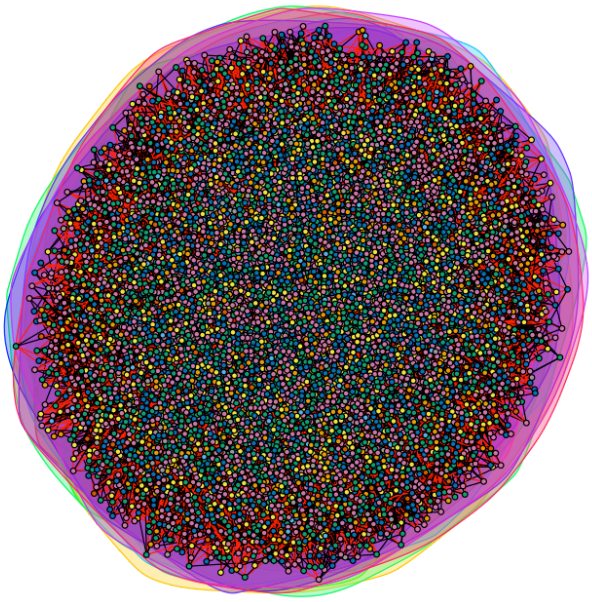


Figure: Community Structures in Preferential Attachment Network (n=10000, m=5)

Community sizes:
[381   75 1891   64   79 171 1622 233 356   36 292 963 1859   1971]

The modularity = 0.270160243631759, which is smaller compared to the smaller network's modularity. It is lower than the modularity of m = 1 and 2. Modularity measures the division of the subgraph. Based on the calculation equation of modularity shown below

$$Q=1/(2m) * sum((Aij-ki*kj/(2m)) delta(ci,cj),i,j)$$

m is the the edges number of the graph. A is adjacency matrix. k is degree. c is the component of a node. delta is i == j. With more edges it is naturally to have smaller modularity. It also follows intuition. When a new node tries to attach to more nodes, it may become a connection between different communities and thereby reduce the modularity of the whole graph.

We plotted the degree distribution in a log-log scale for both n = 1000, 10000; then estimated the slope of the plots:

n = 1000:

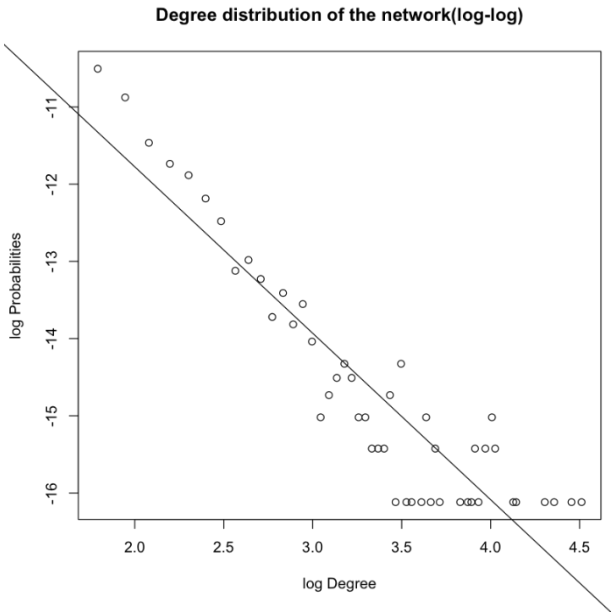


Figure: Degree Distribution in a Log-Log Scale (n=1000, m=5)

By linear regression, we have:

$$log\ (Probability) = -7.463 - 2.156\ log(Degree)$$

The estimated slope is -2.156

n=10000:

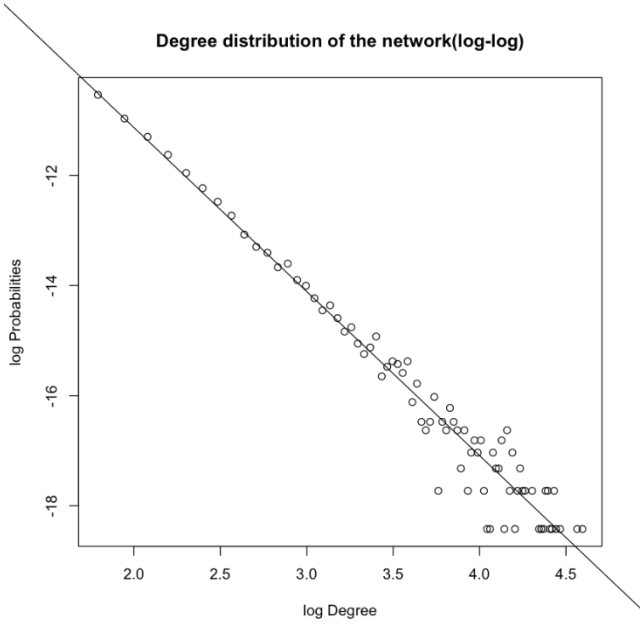


Figure: Degree Distribution in a Log-Log Scale (n=10000, m=5)

By linear regression, we have:

$$log\ (Probability) = -5.171 - 2.979\ log(Degree)$$

The estimated slope is -2.979

Next, we randomly pick a node i, and then randomly pick a neighbor j of that node. We plot the degree distribution of nodes j that are picked with this process, in the log-log scale:

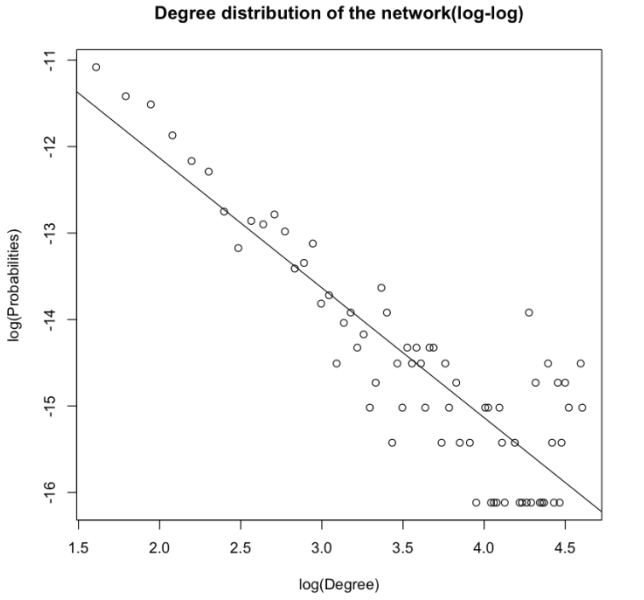


Figure: Degree Distribution of Nodes j in a Log-Log Scale (n=10000, m=5)

By linear regression, we have:

$$log\ (Probability) = -9.129 - 1.502\ log(Degree)$$

The estimated slope is -1.502

Compared to the results of the previous part, the absolute value of slope of this plot is much smaller.

We then want to estimate the expected degree of a node that is added at time step i for $1 \leq \square \leq 1000$. The relationship between the age of nodes and their expected degree is shown below:

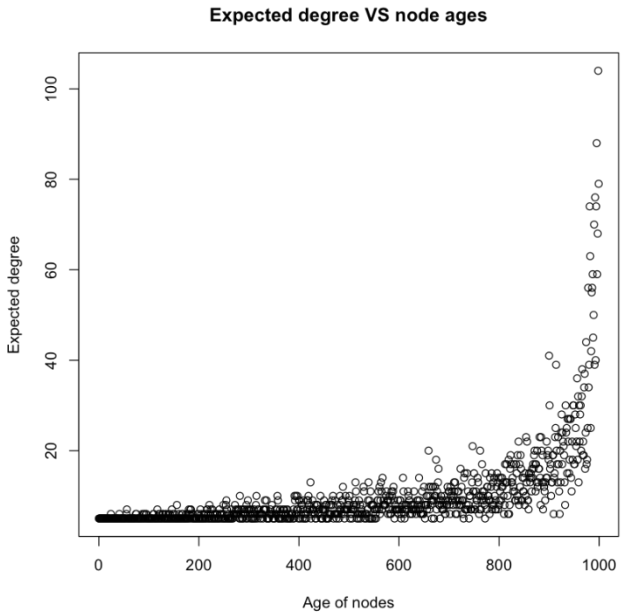


Figure: Expected Degree vs Age of Nodes (m=5)

The reason why modularity for $m = 1$ is high is that: According to the definition of preferential attachment model, small m tends to result in dense connections between the vertices within clusters but sparse connections between vertices of different clusters, which corresponds to a high modularity.

Part (h)

Again, we generated a preferential attachment network with n =1000, m = 1. We took its degree sequence and created a new network with the same degree sequence, through stub-matching procedure.
Below, we plotted both networks and marked the communities:

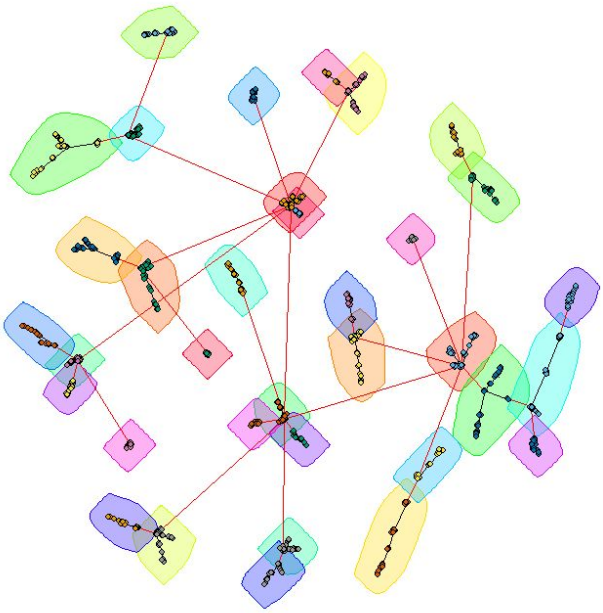


Figure: Community Structures of the Network Generated Using Preferential Attachment Model

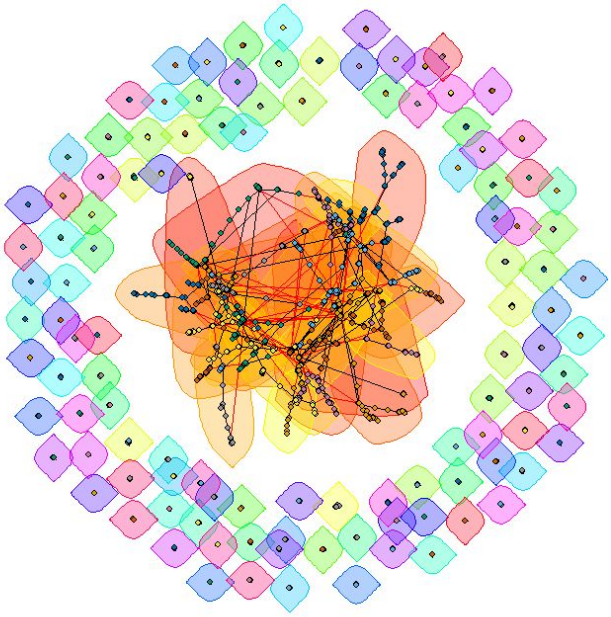


Figure: Community Structures of the Network Generated Using the Reconstructed Model

The modularity of the network generated using preferential attachment model (n=1000, m=1) is 0.932936439943449; The modularity of the reconstructed network is 0.849261173084998.

The reconstructed graph based on stab matched method which does not guarantee the connection of the whole graph. Therefore, more edges are concentrated on the middle graph and thereby result in a lower modularity.

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

Part (a)

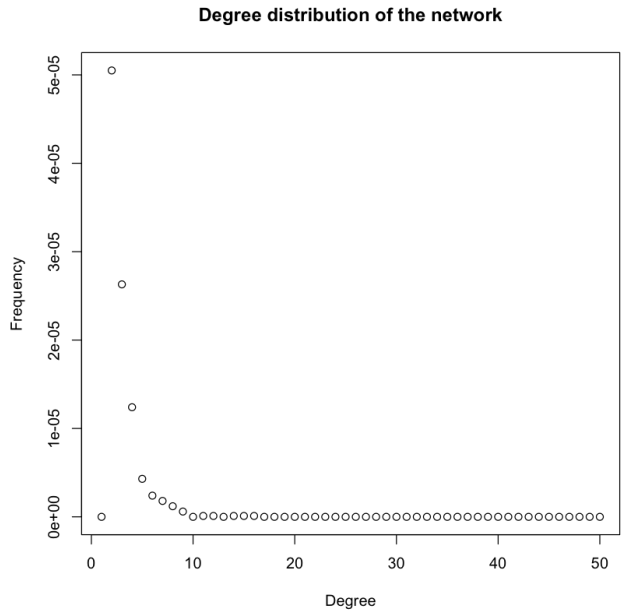


Figure: Degree Distribution of Network Using Modified Preferential Attachment Model that Penalizes the Age of a Node

We plotted the degree distribution in a log-log scale:

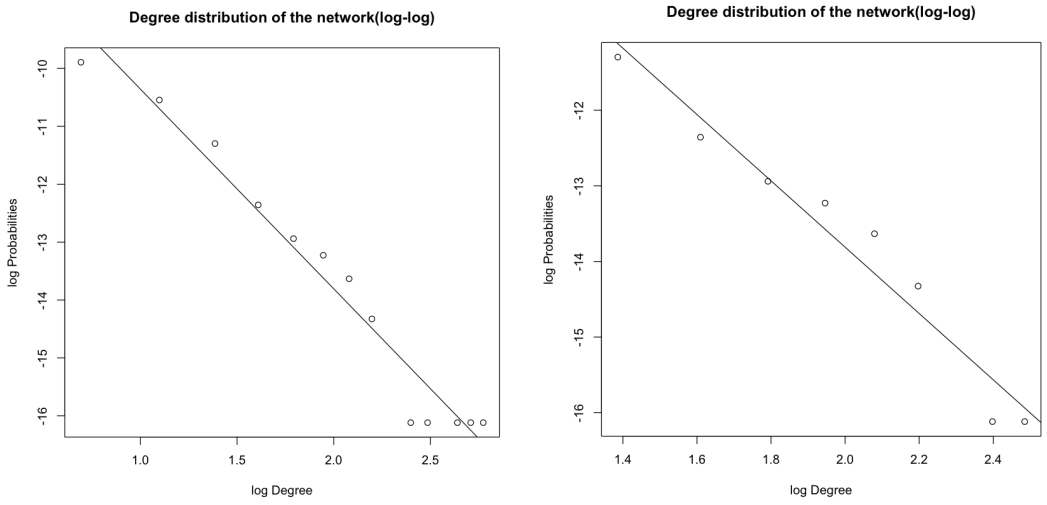


Figure: Degree distribution of the graph:
Left:before truncation Right:after truncation

By linear regression, we have:

$$log\ (Probability)\ = -\ 6.904 - 3.452\ log(Degree)$$

The estimated slope is -3.452.

However, it does not follow our theory about this model which $P \propto \frac{C}{K^5}$. It may caused by the noise and outlier of the dataset. After truncated the initial and ending data point, we could get a model of

$$log\ (Probability)\ = -5.040 - 4.385\ log(Degree)$$

The power law exponent is - 4.385 therefore.

Part (b)

We then used fast greedy method to find the community structure.

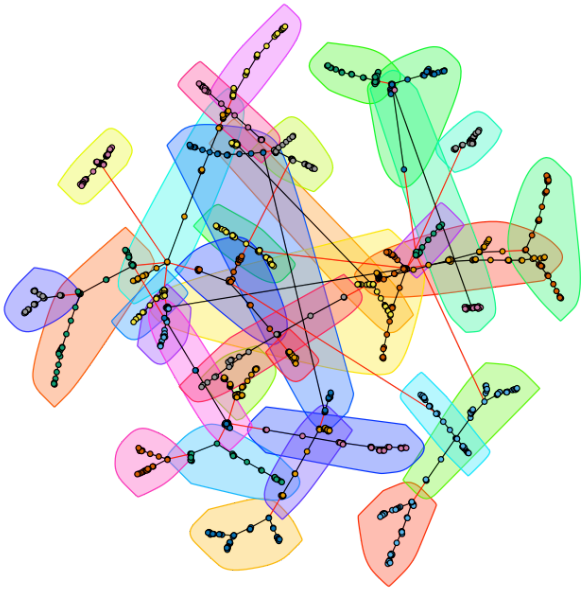


Figure: Community Structures Using Fast Greedy Method

Community sizes:
[43 41 41 41 40 38 40 35 38 35 34 33 32 33 34 35 28 30 27 27 25 27 25 24 24 22 27 28 29 30 31 32 33 22 20 18 22]

The modularity is 0.935862789716648

2. Random Walk on Networks

2.1 Random walk on Erdös-Rényi networks

Part (a)

We created 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 find that the network is connected, and the diameter is 5.

Part (b)

We let a random walker start from a randomly selected node (no teleportation). We used t to denote the number of steps that the walker has taken. We measured the average distance <s(t)>. Also, we measured the standard deviation of this distance.

Below, we plotted *<s(t)> v.s. t* and *σ(t)^2 v.s. t*. Here, the average < > is over random choices of the starting nodes.

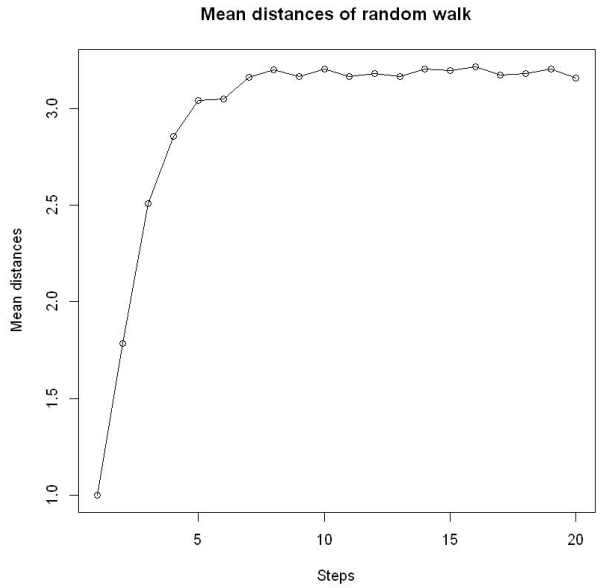


Figure: Mean Distances of Random Walk v.s. Steps in ER Network (n=1000)

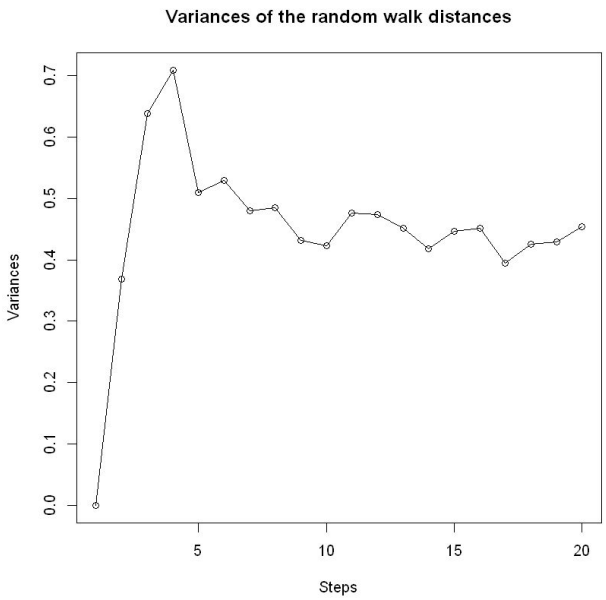


Figure: Variances of the Random Walk Distances v.s. Steps in ER Network (n=1000)

We can observe that the mean distances are upper bounded by the diameter, and number of steps to reach the plateau region of the mean and variance is also affected by the graph diameter.

Part (c)

We then measured the degree distribution of the nodes reached at the end of the random walk and derived the two plots below. The number of steps is 20 for this question.

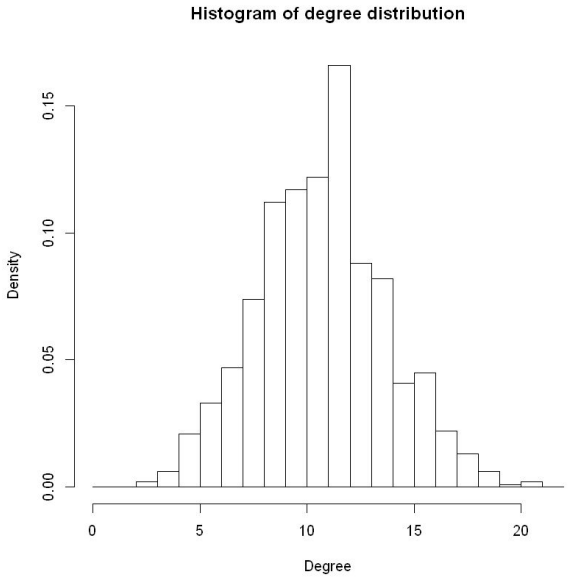


Figure: Degree Distribution of the Nodes Reached at the End of the Random Walk in ER Network (Histogram)

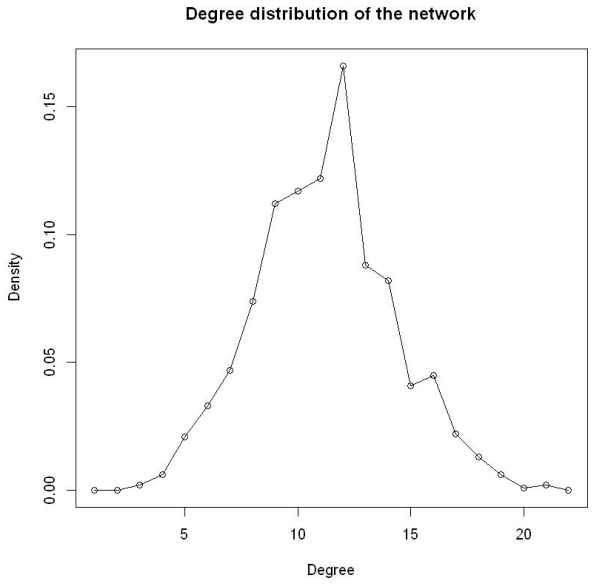


Figure: Degree Distribution of the Nodes Reached at the End of the Random Walk in ER Network

We want to compare the degree distribution of the nodes reached at the end of the random walk to the degree distribution of graph.

Below we derived the plots of the degree distribution of graph:

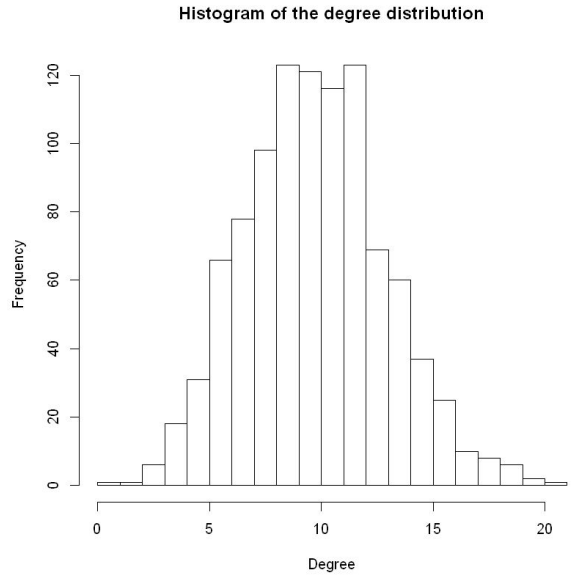


Figure: Degree Distribution of the ER Network (histogram)

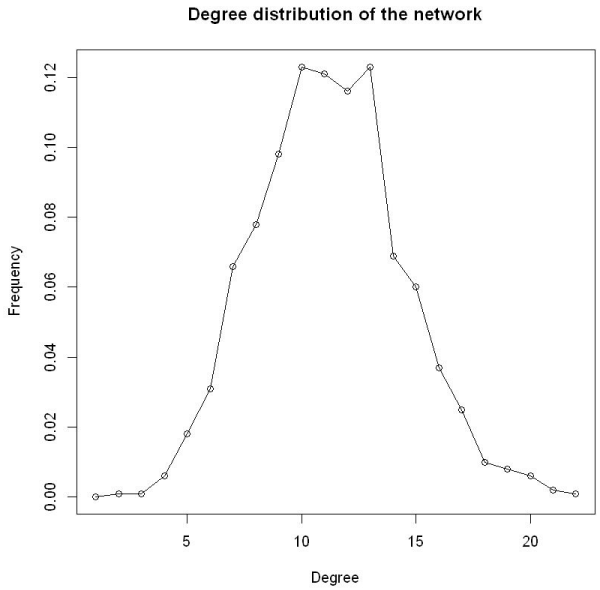


Figure: Degree Distribution of the ER Network

Compared to the degree distribution of the graph, the degree distribution of nodes reached at the end of the random walk is similar in overall shape, with minor difference near the area of peak. This clear peak may imply that the random walk tends to select nodes with higher degrees.

Part (d)

We repeat part (b) for undirected random networks with 100 and 10000 nodes.

Results of n=100:

The network is not connected. The diameter of GCC is 12. Since the graph is not connected, we choose to walk randomly on the GCC only.

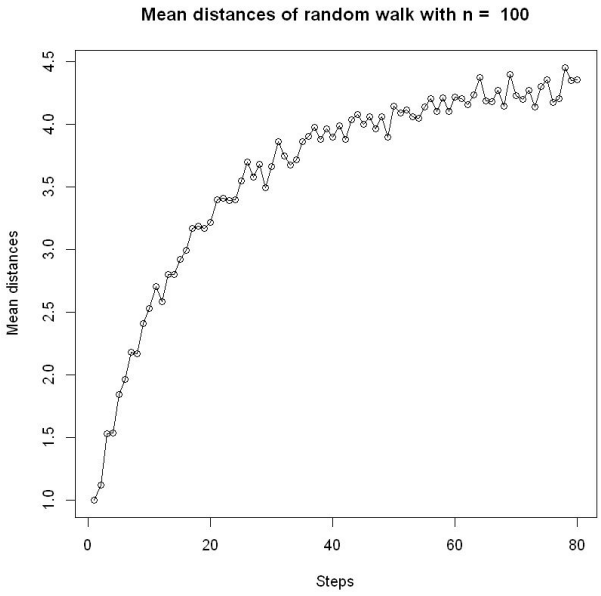


Figure: Mean Distances of Random Walk v.s. Steps in ER Network (n=100)

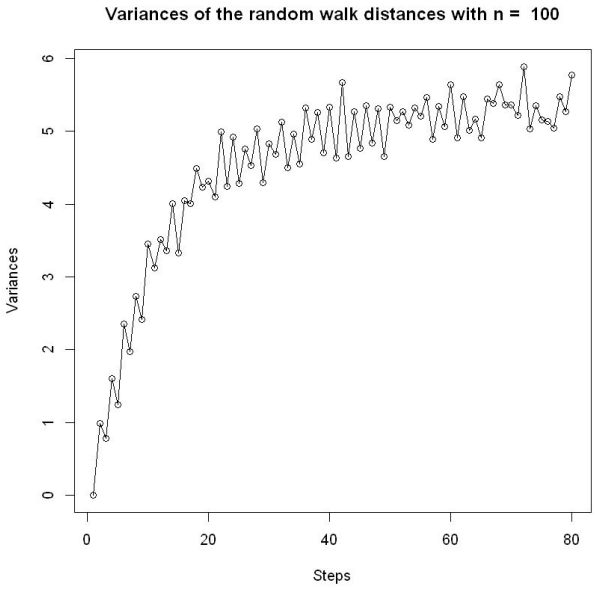


Figure: Variances of the Random Walk Distances v.s. Steps in ER Network (n=100)

Results of n=10000:

The network is connected. The diameter of GCC is 3.

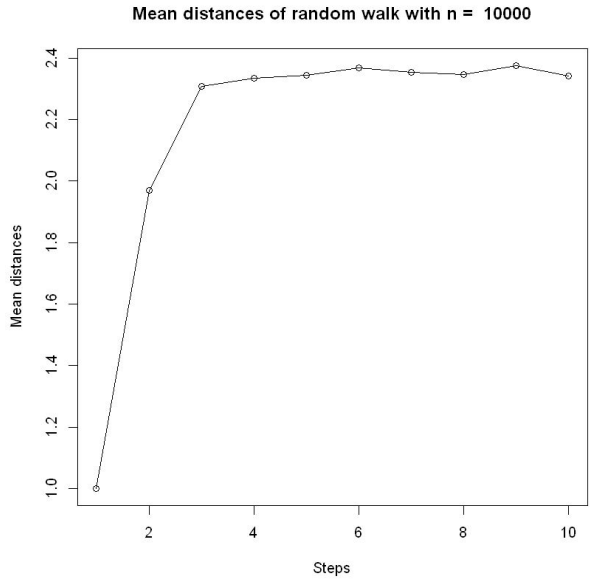


Figure: Mean Distances of Random Walk v.s. Steps in ER Network (n=10000)

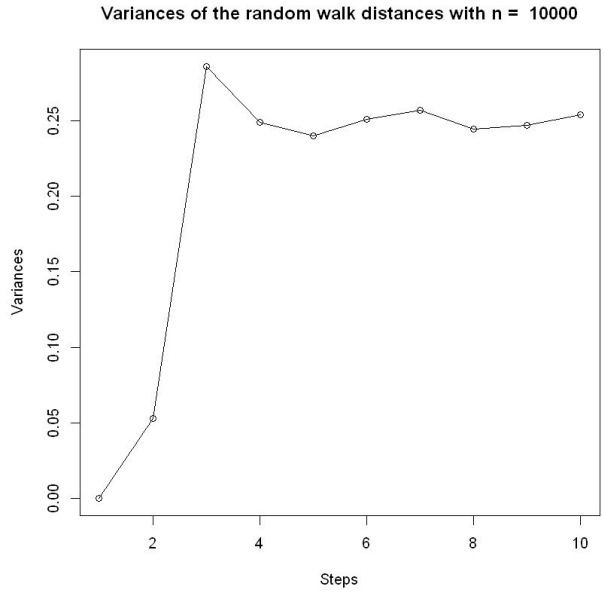


Figure: Variances of the Random Walk Distances v.s. Steps in ER Network (n=10000)

We can see that the diameter of the graph is affected by the number of nodes, i.e. the graph with fewer number of nodes will have a larger diameter in general based on the observation. The diameter would affect the number of steps required to reach the plateau regions of the mean and variance. The general trend is that larger diameters will lead to more steps to reach the plateau regions. It can also be observed that the mean and variance of the graph with a larger diameter is larger than the mean and variance of the graph with a smaller diameter, yet all means are upper bounded by the diameter. Besides, the plateau regions of the graph with a smaller diameter tend to be more stable (with less fluctuation compared with the graph with a larger diameter) under the same number of iterations for averaging.

2.2 Random walk on networks with fat-tailed degree distribution

Part (a)

We generated an undirected preferential attachment network with 1000 nodes, where each new node attaches to m = 1 old nodes. The diameter of the graph is 19.

Part (b)

We let a random walker start from a randomly selected node and measured and plotted $\langle s(t) \rangle$ v.s. t and $\sigma(t)^2$ v.s. t.

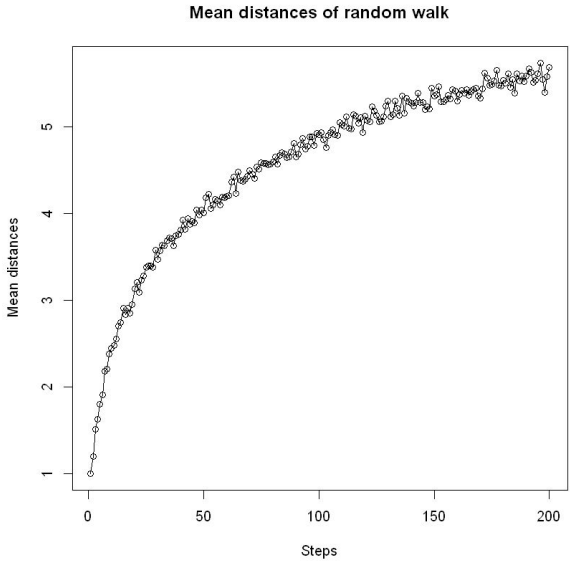


Figure: Mean Distances of Random Walk v.s. Steps in Preferential Attachment Network (n=1000)

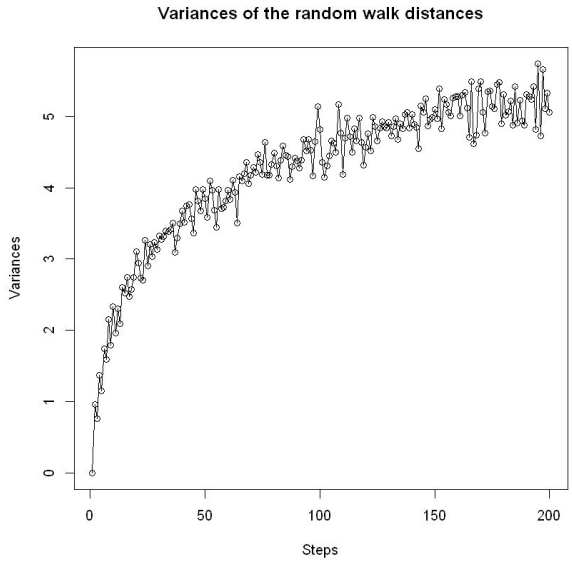


Figure: Variances of the Random Walk Distances v.s. Steps in Preferential Attachment Network (n=1000)

Similar to 2.1(a), we can observe that the mean distances are upper bounded by the diameter, and number of steps to reach the plateau region of the mean and variance is affected by the graph diameter. Besides, since the diameter of this graph is too large, we set the number of steps to be 20.

Part (c)

We then measured the degree distribution of the nodes reached at the end of the random walk and derived the two plots below. The number of steps is 200 for this question since the diameter of the graph is too large.

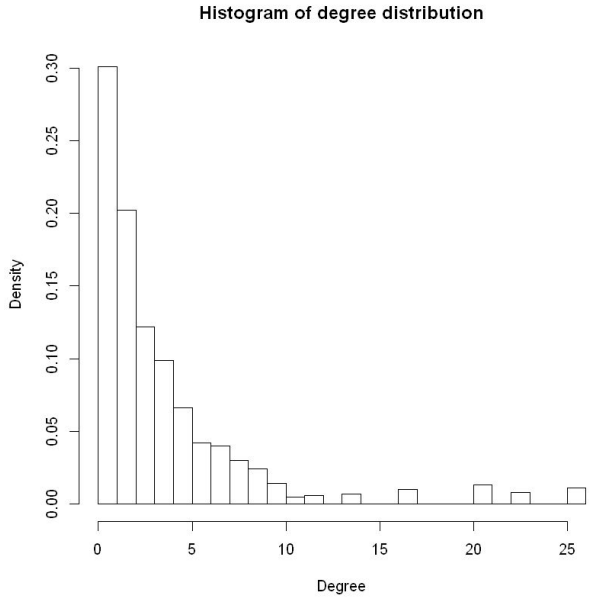


Figure: Degree Distribution of the Nodes Reached at the End of the Random Walk in Preferential Attachment Network (Histogram)

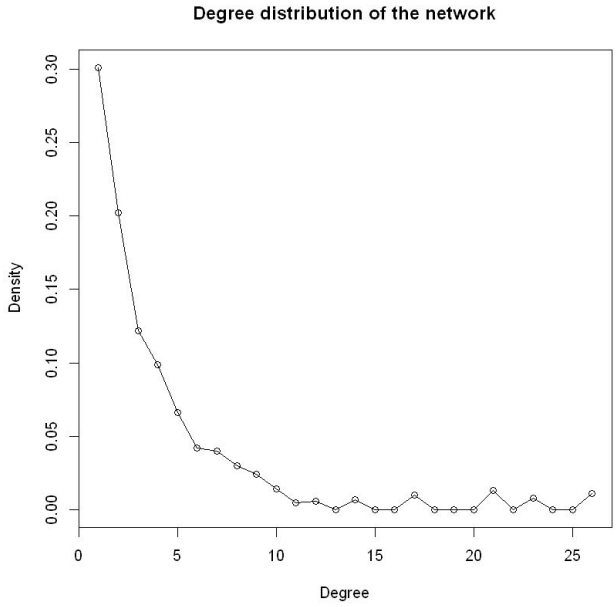


Figure: Degree Distribution of the Nodes Reached at the End of the Random Walk in Preferential Attachment Network

We plotted the distribution in a Log-Log Scale:

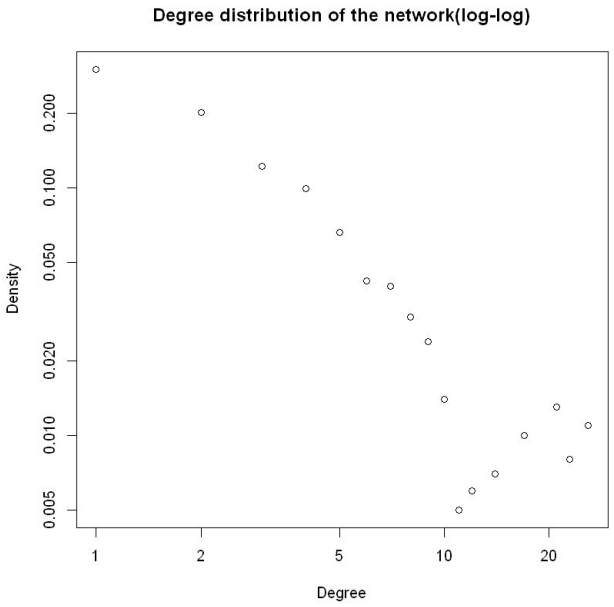


Figure: Degree Distribution of the Nodes Reached at the End of the Random Walk in Preferential Attachment Network (Log-Log Scale)

We would like to compare the degree distribution of the nodes reached at the end of the random walk to the degree distribution of the graph.

Below we derived the plots of the degree distribution of graph:

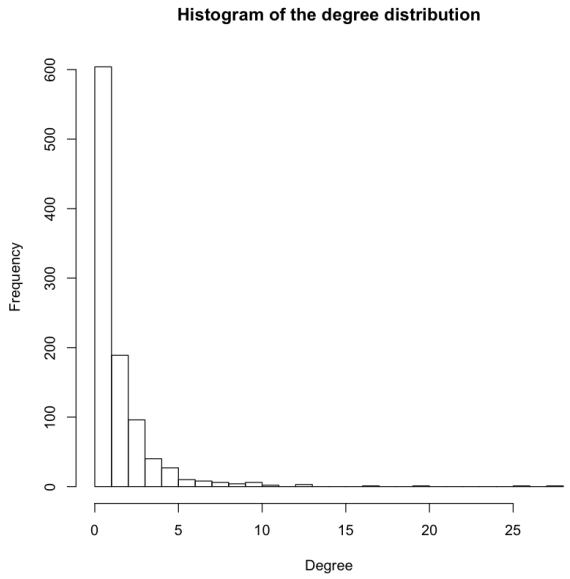


Figure: Degree Distribution of the Preferential Attachment Network (histogram)

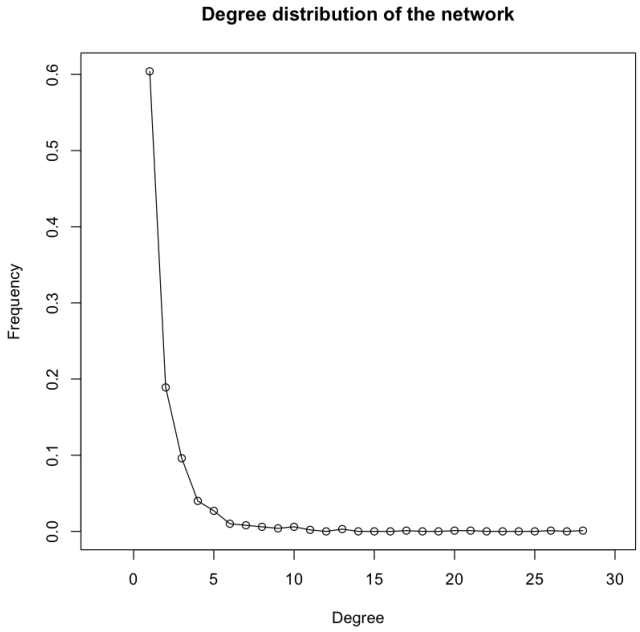


Figure: Degree Distribution of the Preferential Attachment Network

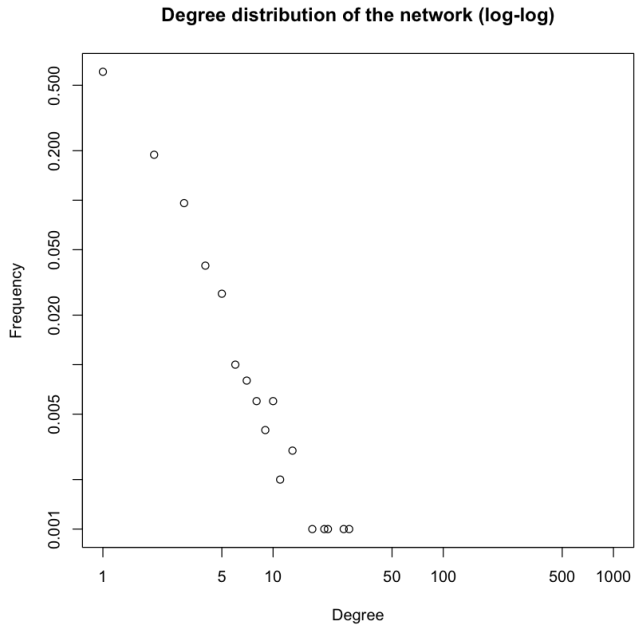


Figure: Degree Distribution of the Preferential Attachment Network (Log-Log Scale)

Compared to the degree distribution of the graph, the degree distribution of nodes reached at the end of the random walk is similar in overall trend, with minor difference at the marginal part. Both the degree distributions follow the power law distribution as shown in the log-log graph, however, based on the observation, the absolute value of the slope in the random walk graph is smaller than absolute slope in the degree distribution graph. The potential reason could be the number of steps we used is insufficient in this case (due to the limited computation, we use 200 steps,which might be relatively smaller for a large diameter graph ~20). Thereby the walker may

not end in the steady state and this result may not reflect the actual random walk distribution at the steady state.

Part (d)

We then repeated part (b) for preferential attachment networks with 100 and 10000 nodes, keeping m = 1.

Results of n=100:

The network is connected. The diameter of GCC is 12.

Figure: Mean Distances of Random Walk v.s. Steps in Preferential Attachment Network (n=100)

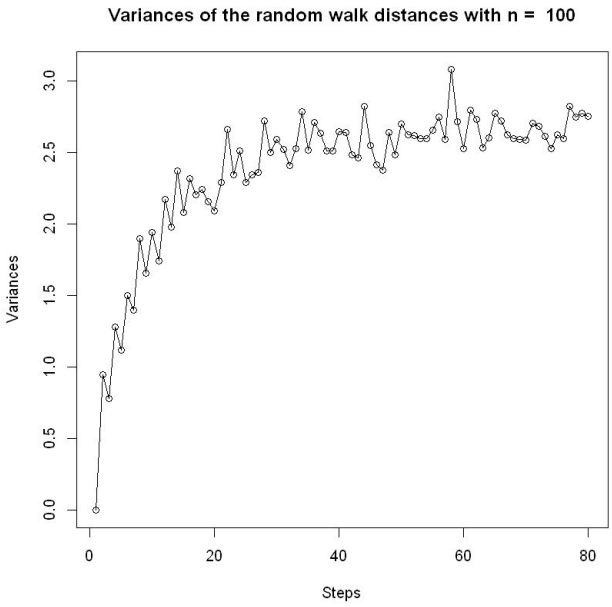


Figure: Variances of the Random Walk Distances v.s. Steps in Preferential Attachment Network (n=100)

Results of n=10000:

The network is connected. The diameter of GCC is 12.

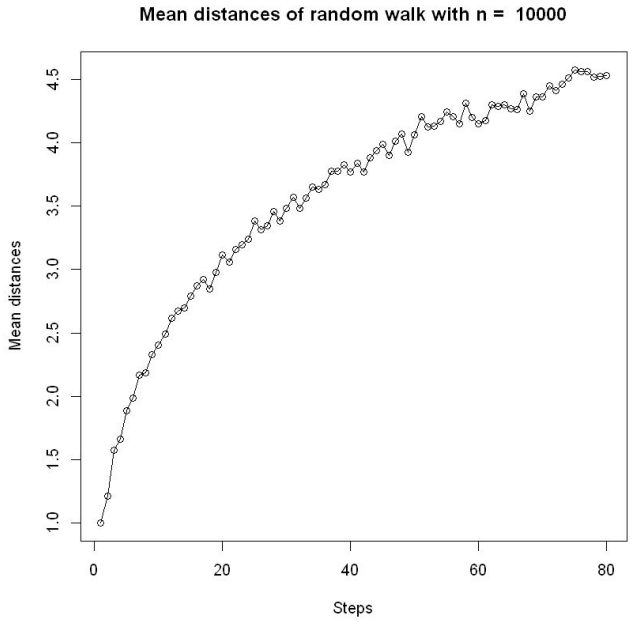


Figure: Mean Distances of Random Walk v.s. Steps in Preferential Attachment Network (n=10000)

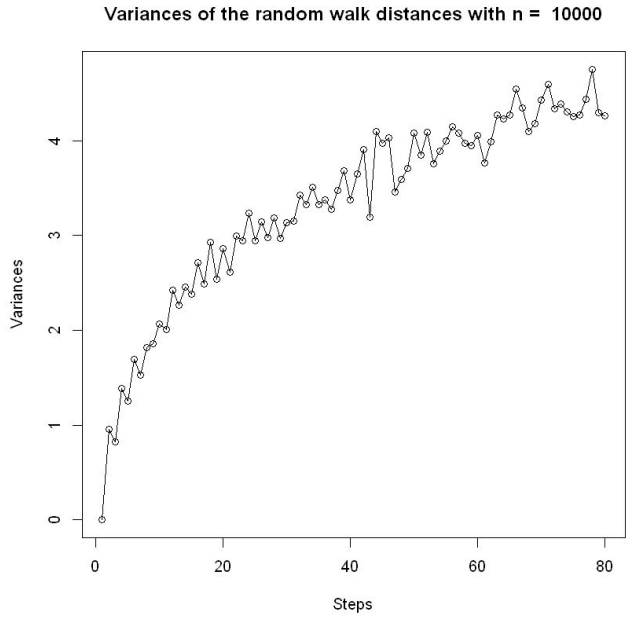


Figure: Variances of the Random Walk Distances v.s. Steps in Preferential Attachment Network (n=10000)

Again, similar phenomenon is observed. The diameter would affect the number of steps required to reach the plateau regions of the mean and variance. The general trend is that larger diameters will lead to more steps to reach the plateau regions. Besides, the plateau regions of the graph with a smaller diameter tend to be more stable (with less fluctuation compared with the graph with a larger diameter) under the same number of iterations for averaging.

2.3 PageRank

Note: To reach the steady states, we choose to walk 20 steps for both 2.3 and 2.4.

Part (a)

The probability that walker visits each node and the in-degree distribution are as follows.

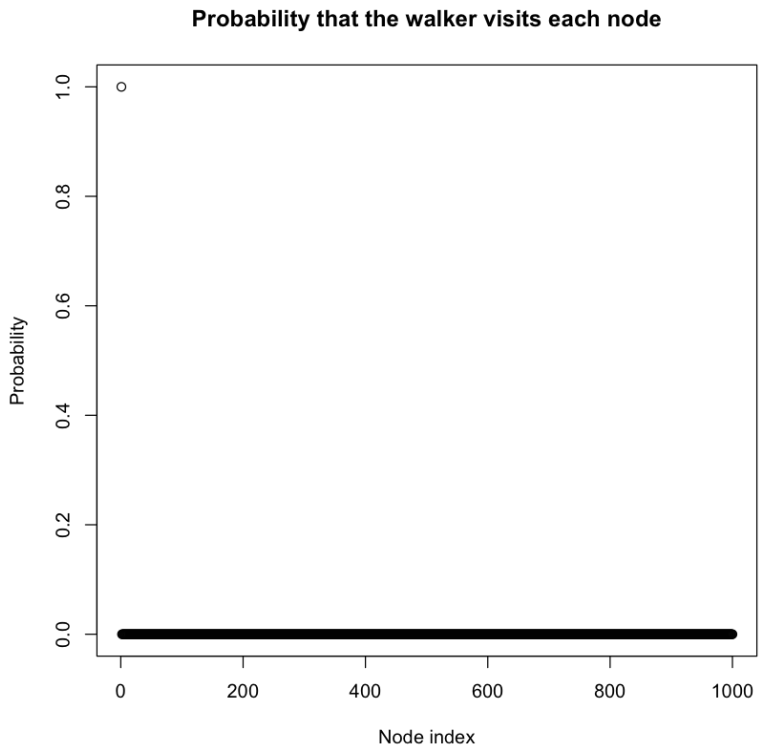


Figure: Probability that the walker visits each node (m=4, n=1000)

It can be seen that the first node has probability 1 while all others have probability 0. This is because the number of steps is large enough and thus random walk reaches a steady state. Considering the fact that the newly-added vertices will have directed edges to the existing node, no matter what the start point is the random walk will always end at the first node.

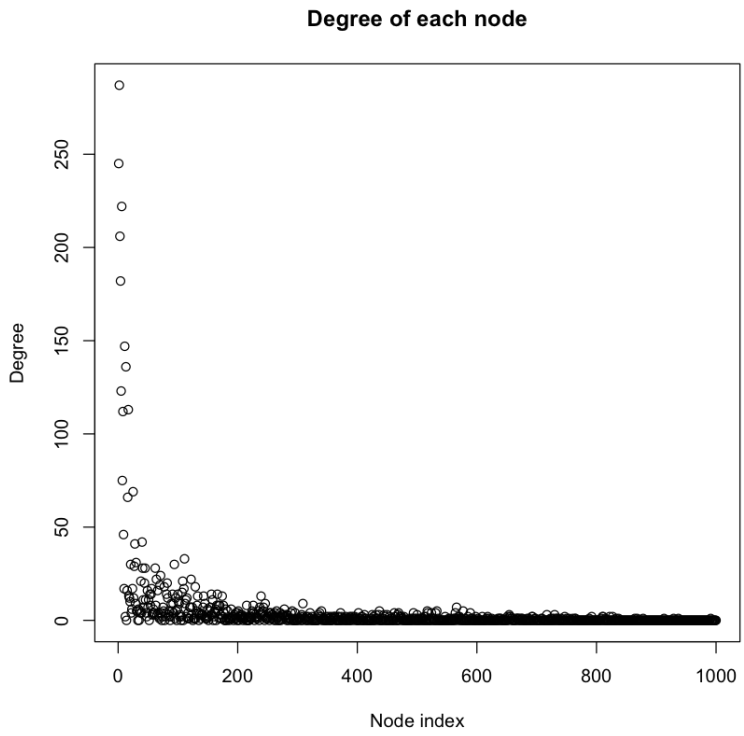


Figure: Degree distribution of the network

Correlation between visiting probability and graph in-degree: 0.5674746, so there's no strong relation between them.

Part (b)

First we derive the transition matrix.
Taking into account the teleportation, one can write ***PR*** as

$$\mathbf{PR} = (1\text{-}d)/N \times \mathbf{1} + dA \times \mathbf{PR}$$

where **1** is the column vector of all 1's.
And **1** can be written as

$$\mathbf{1} = \mathbf{[1]} \times PR$$

where **[1]** denotes the n *n matrix with all elements equal to 1.
Therefore,

$$\mathbf{PR} = [(1\text{-}d)/N \times \mathbf{[1]} + dA] \times \mathbf{PR}$$

And the transition matrix is

$$(1\text{-}d)/N \times \mathbf{[1]} + dA$$

The probability that walker visits each node is as follows.

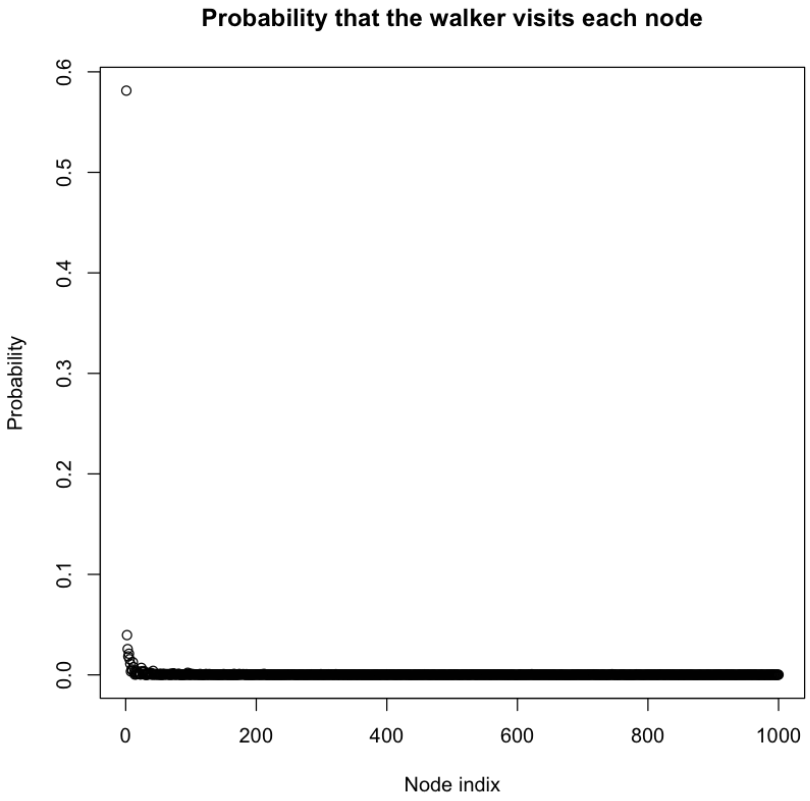


Figure: Probability that the walker visits each node (m=4, n=1000)

Compared to 2.3(a), it can be seen that the probability of visiting the first vertex is no longer 1. Some other vertices are visited, too. This is because with the teleportation term, the walker will be able to visit a random node which is not connected to it. Therefore, even when the walker reaches the first node, it still has chances to go to other nodes.

The Correlation coefficient with the degree of graph g is 0.6391381, so there's no strong relation between them.

2.4 Personalized PageRank

Part (a)

First, we derived the new transition matrix based on the following equation of the PageRank.

$$\boldsymbol{PR} = (1\text{-}d)\boldsymbol{PR} + dA\boldsymbol{PR}$$

where d is the teleportation probability, $\boldsymbol{PR}$ is the PageRank vector, and A is the original transition matrix.

By utilizing the the idea that $\boldsymbol{PR} = \mathbf{I}\times\boldsymbol{PR}$, where $\mathbf{I}$ is the identity matrix, we can obtain the following equation.

$$\boldsymbol{PR} = (1\text{-}d)\mathbf{I}\boldsymbol{PR} + dA\boldsymbol{PR} = [(1\text{-}d)\mathbf{I} + dA]\boldsymbol{PR}$$

So, the new transition matrix can be written as (1-d)**I** + dA. We use this new transition matrix for random walking and the following results are obtained.

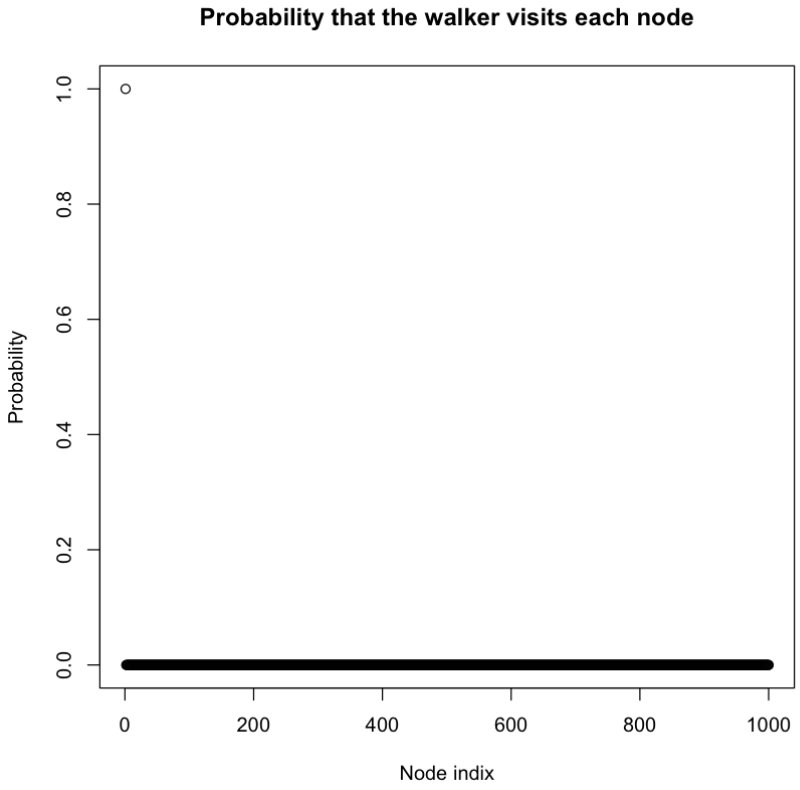


Figure: Probability that the walker visits each node

Correlation with degree(g): 0.5674746

From the above figure, we know the result of 2.4(a) is exactly the same as the result of 2.3(a). This is rational since the the transition matrices of these two settings are the same to some extent. The transition matrix in 2.4(a) just further incorporates the chance that the walker stays in the same node during walking. The actual steady states of the random walking with these two settings should be the same, except that with transition matrix of 2.4(a), it may take more steps to converge due to the self loops. This phenomenon can be reviewed with the following equations as well, where we can observe that the original A is also a suitable transition matrix for this ***PR***.

$$\boldsymbol{PR} = (1\text{-d})\boldsymbol{PR} + \text{dA}\boldsymbol{PR} \Rightarrow \boldsymbol{PR} = \text{A}\boldsymbol{PR}$$

Part (b)

In this section, we restricted teleportation to the node with median PageRank. More specifically, in each step, there is 15% probability that the walker jumps to one of the two median nodes, and there is 85% chance that the walker follows the original transition matrix. In addition, since the PageRank is proportional to the age of the nodes, we choose the 500 and 501 nodes in the graph for teleportation. Therefore, the teleportation vector is a vector of all zeros, except that the values in position 500 and position 501 are 0.5 respectively.The random walking results are shown below.

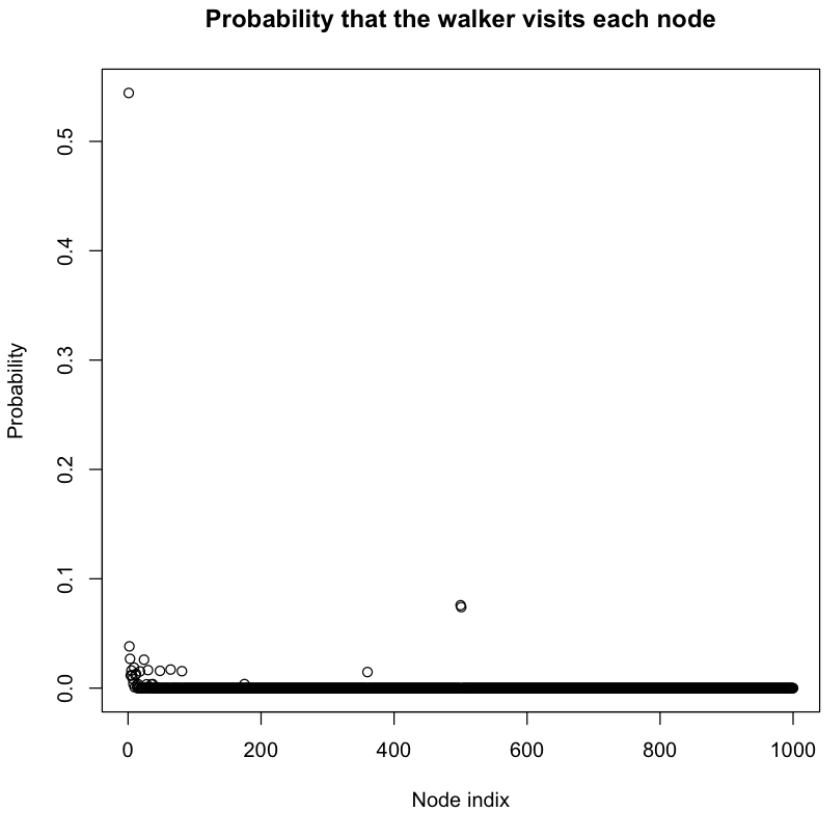


Figure: Probability that the walker visits each node

Correlation with degree(g): 0.6333448

As expected, the probability distribution in 2.4(b) differs from the distribution in 2.4(a). The walker may not always stop at the first node as 2.4(a). Instead, there is chances that the walker stops at some other early nodes and the node 500 and 501. Besides, since the correlation is 0.6333448, there is no strong relationship between this probability distribution and the degree distribution.

Part (c)

In this part, we aim to drive the transition matrix which incorporates this self-reinforcement. The new transition matrix can be deducted as follows.

PR can be written as

$$\boldsymbol{PR} = (1\text{-}d)W + dA\boldsymbol{PR}$$

where W is the teleportation vector shown below, w is the number of trusted web pages.

$$\left[0 \quad \ldots \quad 0 \quad \frac{1}{w} \quad \ldots \quad \frac{1}{w} \quad 0 \quad \ldots \quad 0\right]^T$$

Further, W can be represented as

$$W = C \times \boldsymbol{PR}$$

where C is

$$\begin{bmatrix} 0 & \cdots & 0 \\ \vdots & & \vdots \\ \frac{1}{w} & \cdots & \frac{1}{w} \\ \vdots & & \vdots \\ \frac{1}{w} & \cdots & \frac{1}{w} \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{bmatrix}$$

Therefore, substituting W leads to

$$\boldsymbol{PR} = [(1\text{-}d)C + dA]\boldsymbol{PR}$$

and the transition matrix is (1-d)C + dA.

This equation represents the process of self-reinforcement. The trusted pages are more easily to get higher pagerank because of the additional term 1/w. By multiplying the transition matrix multiple times, the pagerank of trusted web pages will continue to grow, and others will shrink. Finally the trusted web pages will dominate the pagerank vector.

As to the implementation, we can use the following codes to generate the new transition matrix. Note: In the codes, the transition matrix is the transpose of the previously derived transition matrix due to different implementation.

```
transition_matrix = create_transition_matrix(g) * (1 - alpha) + repmat(tel_vector, n, 1) * alpha
```