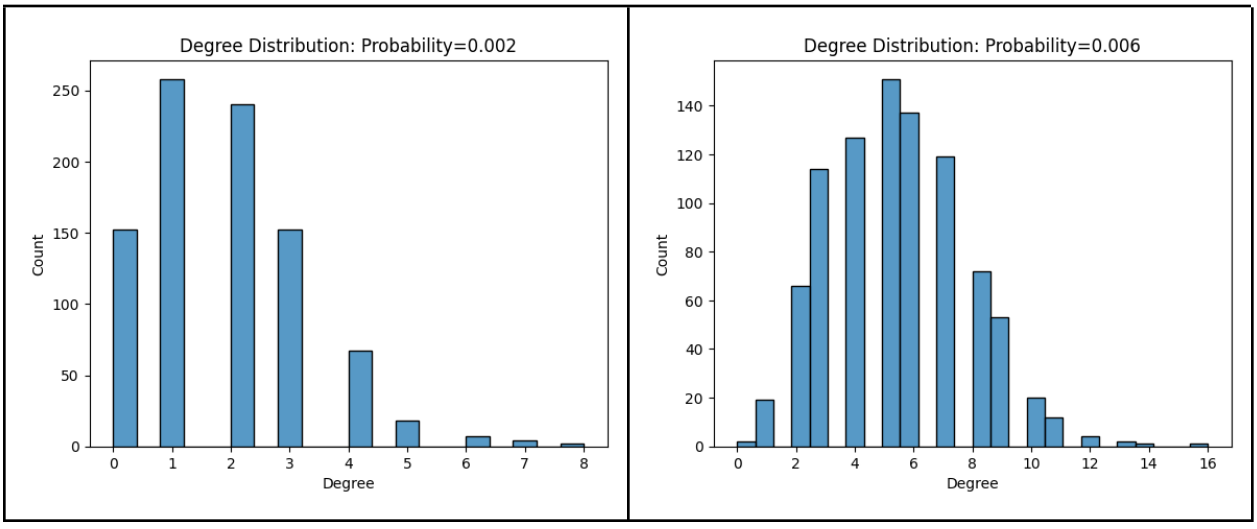
Project 1: Random Graphs and Random Walks

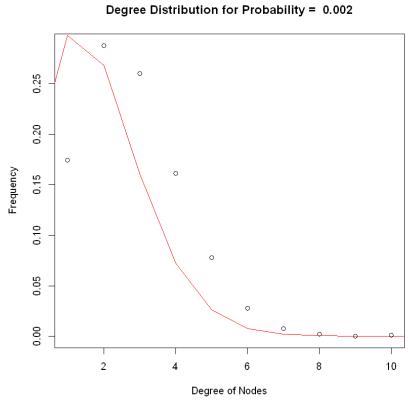
Robert Ozturk, Yaman Yucel, Sarah Wilen

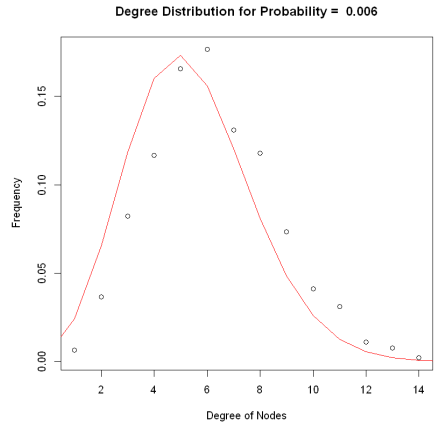
Part 1: Generating Random Networks

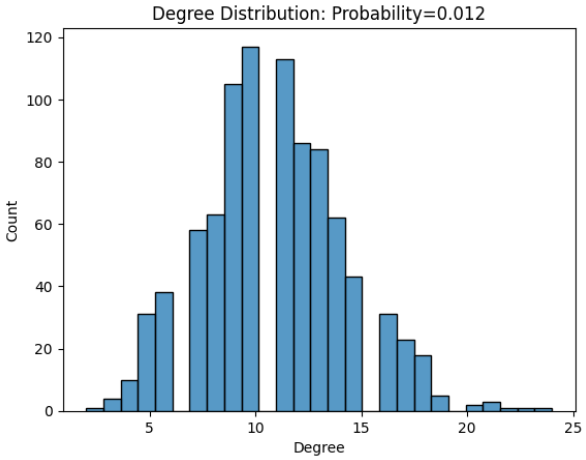
Problem 1: Erdos-Renyi (ER) Model

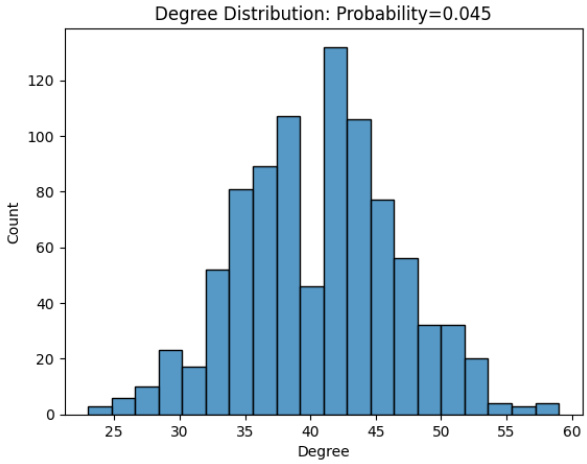
Part a

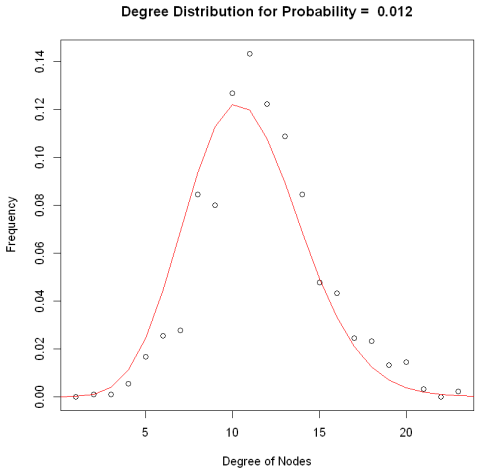


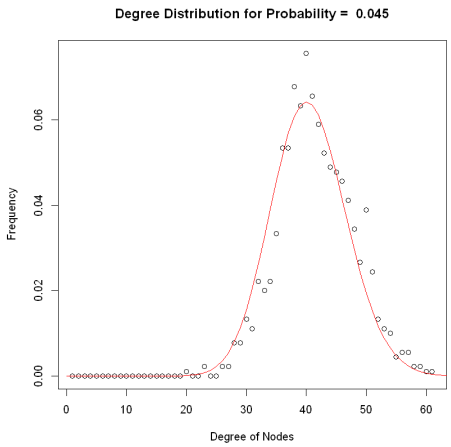

Degree Distribution for Probability = 0.002
Frequency
Degree of Nodes

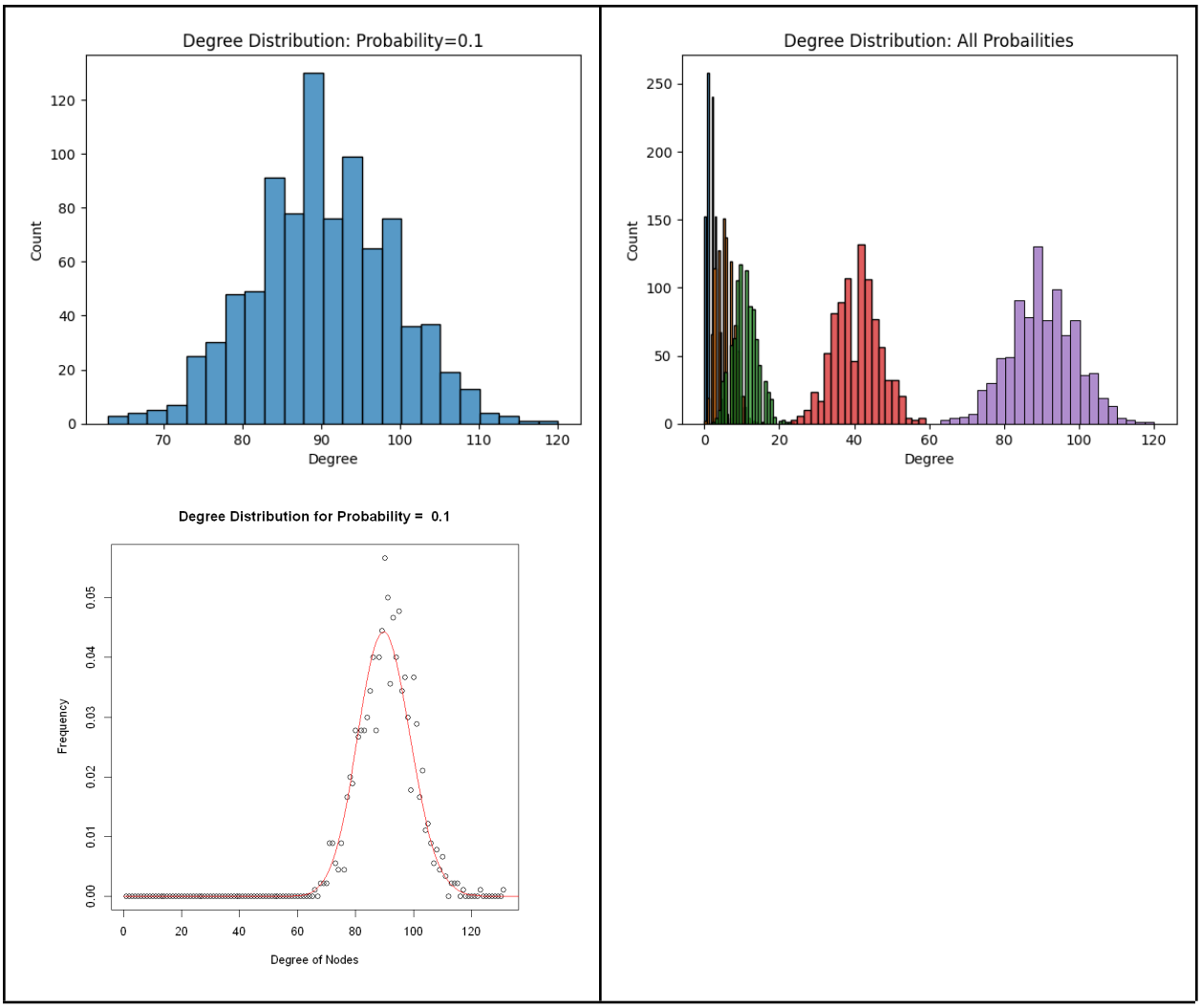

Degree Distribution for Probability = 0.006
Frequency
Degree of Nodes


Degree Distribution: Probability=0.012
Count
Degree


Degree Distribution: Probability=0.045
Count
Degree


Degree Distribution for Probability = 0.012
Frequency
Degree of Nodes


Degree Distribution for Probability = 0.045
Frequency
Degree of Nodes



The **degree distribution is binomial**. This is because the graph is randomly generated. Let d be the degree of a vertex and n be the number of total nodes. There are $\binom{n}{d}$ ways to choose d vertices from n total nodes. The probability that a node has edges to d nodes is therefore p^d. The probability that there are no edges to the rest of the n-d nodes is $(1-p)^{n-d}$. Therefore, we see that $Probability(d) = \binom{n-1}{d} p^d (1-p)^{n-1-d}$, which is a binomial distribution.

The theoretical mean degree of a vertex is calculated as such. Random variable X is an edge. Using the expectation formula: $E[X] = \sum_{for\ all\ x\ in\ X} xPr[X = x] = \sum_{d=0}^{n} d\binom{n}{d}(1-p)^{n-d} = np$

The theoretical variance of degree distribution is calculated by taking the second derivative of the probability distribution. The variance of a binomial distribution is $Var(X) = np(1-p)$

Sample calculation:

$theoretical\ mean = np = 900 * 0.002 = 1.8$

$\%\ difference = |\frac{observed-expected}{expected}| * 100\% = |\frac{1.82-1.8}{1.8}| * 100\% = 1.11\%$

$theoretical\ variance = np(1-p) = 900 * 0.002(1-0.002) = 1.7964$

$\%\ difference = |\frac{observed-expected}{expected}| * 100\% = |\frac{1.892-1.7964}{1.7964}| * 100\%$

Probability (p)	**Observed mean**	**Theoretical mean**	**% error of mean**	**Observed variance**	**Theoretical variance**	**% error of variance**
0.002	1.82	1.8	1.11%	1.892	1.7964	5.3217%
0.006	5.44	5.4	0.823%	5.4459	5.3676	1.85%
0.012	10.911	10.8	1.029%	11.43	10.6704	7.117%
0.045	40.573	40.5	0.181%	36.869	38.677	4.675%
0.1	90.24	90	0.26%	79.856	81	1.143%

The observed mean and variance are very close to the expected mean and variance with on average less than 0.7% error in mean and less than 4% error in variance.

Part b

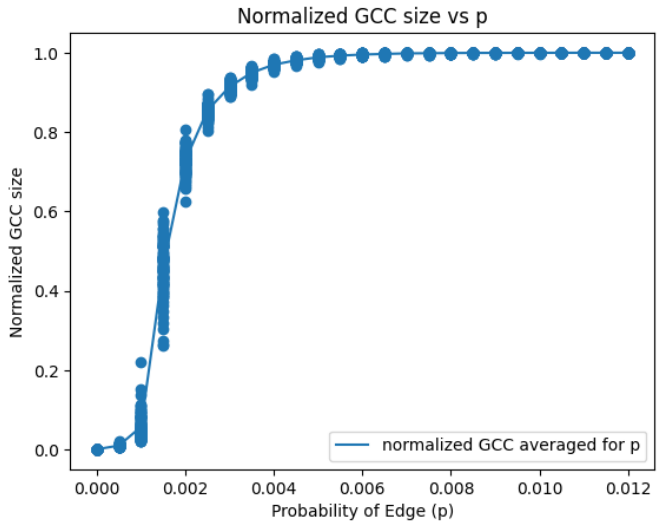
Intuitively, not all random realizations of the ER network are connected because there is only a probability that an edge will be drawn between two arbitrary vertices. Therefore, there might not be a path that exists from every node to each other.

We numerically estimated the probability that a generated network is connected by creating 1000 instances of the ER graph with a specific p and then checking if the network is connected or not.

Probability (p)	**Are all random realizations of the ER network connected?**	**Estimate: probability that a network is connected**	**Diameter of the GCC**
0.002	NO	0	23
0.006	NO	0.018	9
0.012	NO	0.99	5
0.045	YES	1	FULLY CONNECTED
0.1	YES	1	FULLY CONNECTED

As the probability of a connected edge increases, the probability that the network is connected increases and the diameter of the GCC decreases.

Part c



$p_{max} = 0.012$, which makes the network almost surely connected. If $p > \frac{ln(n)}{n}$, then a graph in $G(n, p)$ will almost surely be connected. $0.012 > \frac{ln(900)}{900} = 0.00755$

Part i

A GCC emerges when the normalized GCC starts to increase from 0. From our graph, we see this **occurring around p=0.001.**

The theoretical value is $p = O(\frac{1}{n}) = O(\frac{1}{900}) = 0.001$, which is also very close to what we observed as the value where GCC starts to emerge.

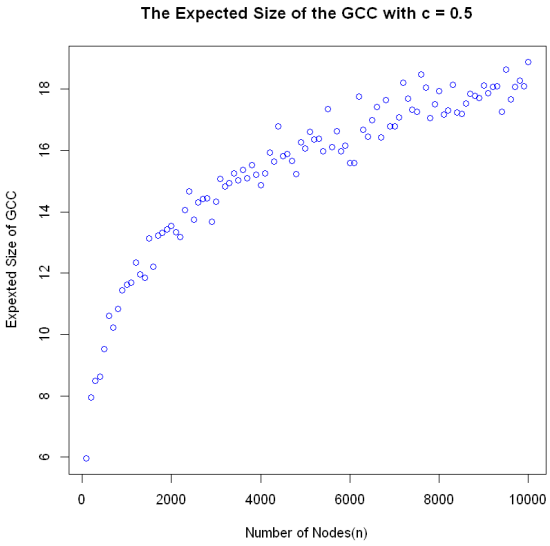
Part ii

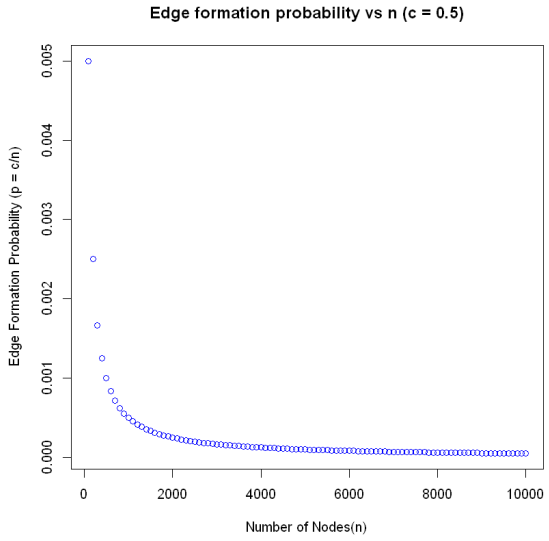
The point where the GCC takes up over 99% of nodes in almost every experiment is where the normalized GCC size approaches 1. From the graph, we can empirically estimate that this point **is around p=0.0065.**

The theoretical value is $p = O(\frac{ln(n)}{n}) = O(\frac{ln(900)}{900}) = 0.00755$, which is very close to what we observed.

Part d

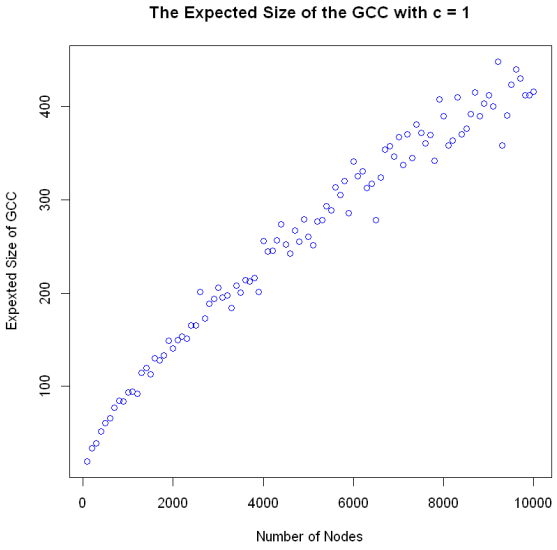
Part i

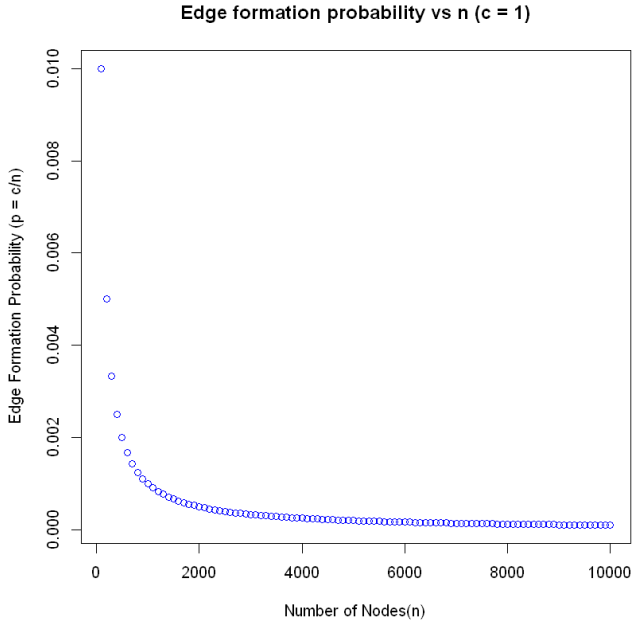




When c=0.5, we notice the expected GCC size increases with increasing number of nodes. At around 2000 nodes, we notice that the GCC size starts to increase linearly. The GCC size stays within the range of 6-20 due to the constant degree of 0.5. If I had averaged out a larger amount of networks per GCC point, I think the scatter plot would resemble more of a linear trend rather than having a lot of variance.

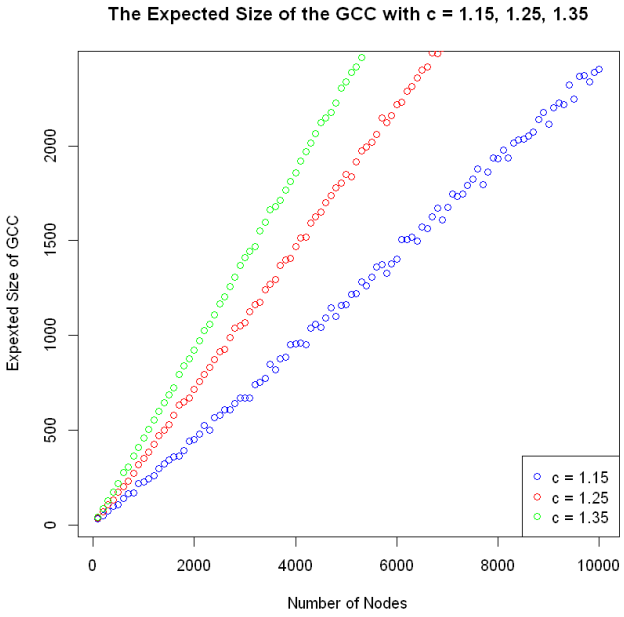
Part ii

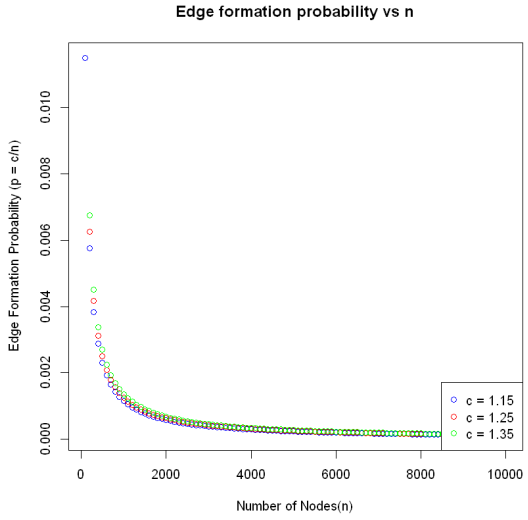




When c=1, we see the same result as in c=0.5 where the GCC size increases with increasing number of nodes. This makes sense as since the average degree of nodes remains the same, but there are more nodes, then the GCC size increases. The range of size of GCC (100-400 nodes) is a lot larger than when c=0.5 because the average degree is higher than before. The trend looks quite linear as well.

Part iii



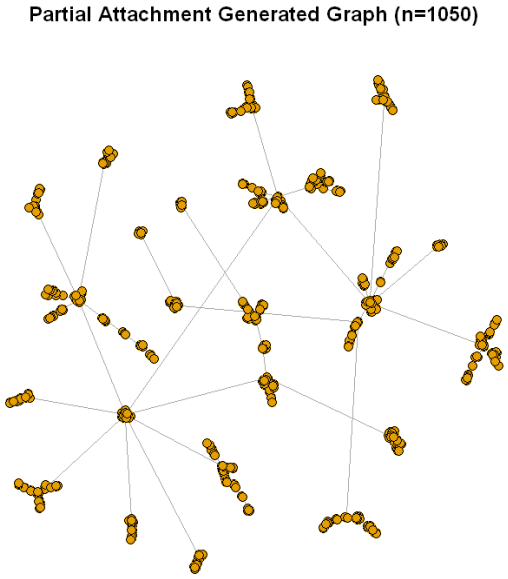


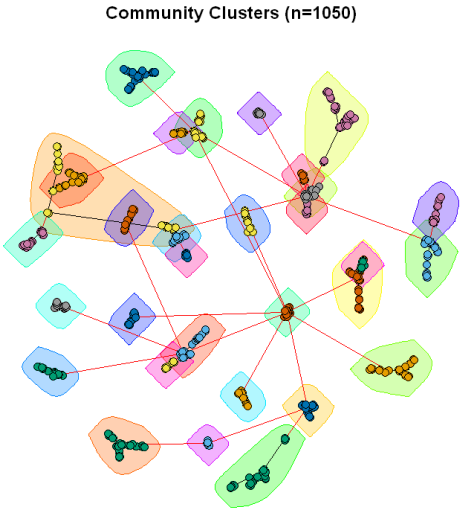
Part iv

Comparing c=1.15, 1.25, 1.35, we see that the higher the c, the steep the slope of the trend. For example, the slope for c=1.35 is a lot stepper than c=1.15. This makes sense because as the c increases, for a given number of nodes, the probability of edge formation (p) increases. Hence, more nodes will likely be connected, so the size of the GCC increases.

Problem 2: Preferential Attachment Model

Part a





Yes, these networks are always connected because they grow by adding a new vertex v to m number of existing nodes. Therefore, there can never be a node added that is not connected to any other nodes.

Part b

Modularity: 0.9241685531001878

Modularity measures the strength of division of a network into modules (clusters). Higher modularity means denser connection between nodes within modules, but sparse connections between nodes in different clusters.

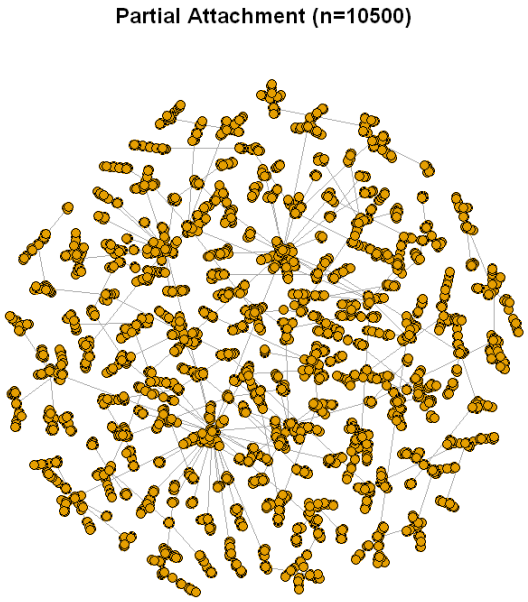
Assortativity refers to how nodes tend to connect with ‘similar’ nodes over ‘dissimilar’ nodes. It measures correlation between vertex properties. Similarity of nodes can be measured in different ways. For example, two nodes can be similar with respect to a property if they have similar values for that property. Weight can be a property on which similarity is measured.

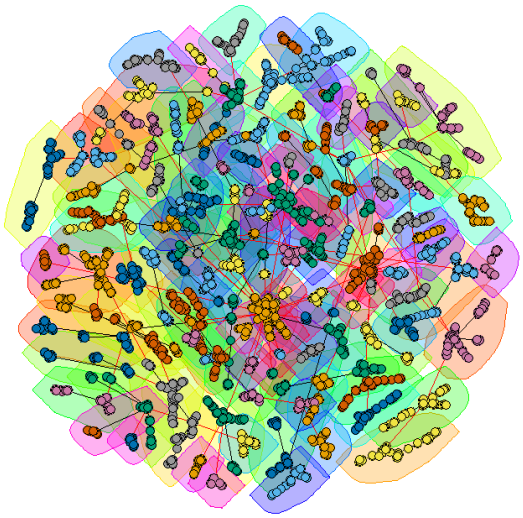
Assortatitivty can be captured by the assortativity coefficient, which is the Pearson correlation coefficient of degree between pairs of linked nodes. Positive values indicate correlation between nodes of similar degree, which means nodes tend to connect to each other.

The measure of **degree assortativity for our graph is: -0.11491808306035368.**

This means, there are more relationships between nodes of different degrees in our network.

Part c





Modularity: 0.9753048882299131
Assortativity Coeff by Degree: -0.0193842287843106

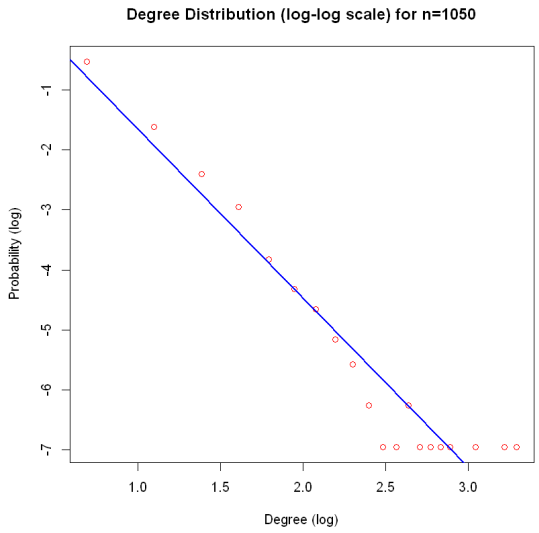
In comparison to 1050 nodes, this graph of 10500 nodes has a slightly higher modularity score meaning there are denser connections between nodes within clusters. Additionally, the degree assortativity for the

graph is not as negative and is almost zero indicating that there are less relationships between nodes of different degrees than in the 1050 node graph. This makes some sense because as the number of nodes increases, the number of nodes in smaller communities increases as well increasing the modularity.

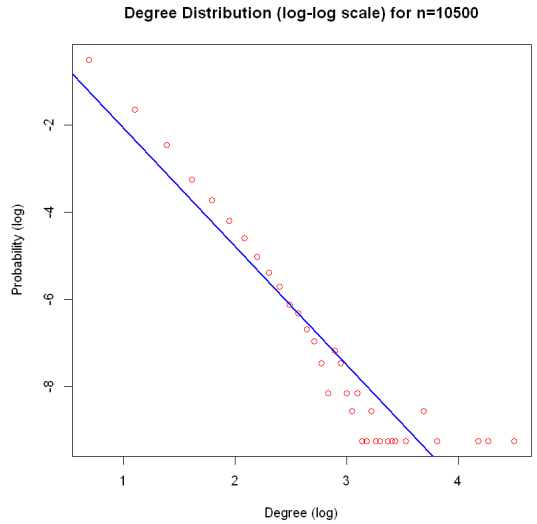
It also has a lower assortativity coefficient, which would suggest that large-degree nodes tend to attach to low-degree nodes, which looks to make sense considering there are even more ndes and since node edge formation probability is proportional to degree, the large degree nodes get even larger by connecting to low-degree nodes.

Part d

n=1050: Slope=-2.815



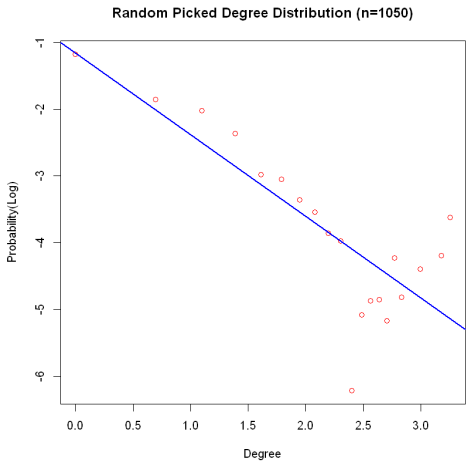
n=10500: slope=-2.7258



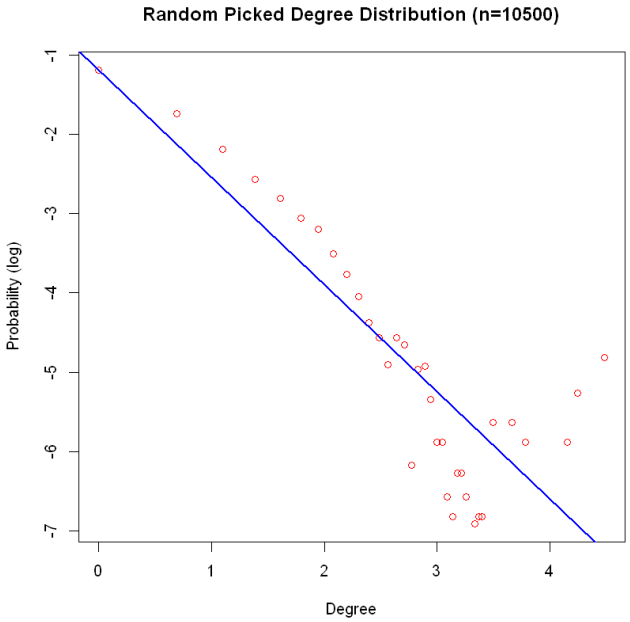
Part e

The distribution is linear in the log-log scale. This differs from the node degree distribution because while the power exponent of the original network is 3 (as expected for a preferential attachment model), this random node selection degree distribution is close to 1.

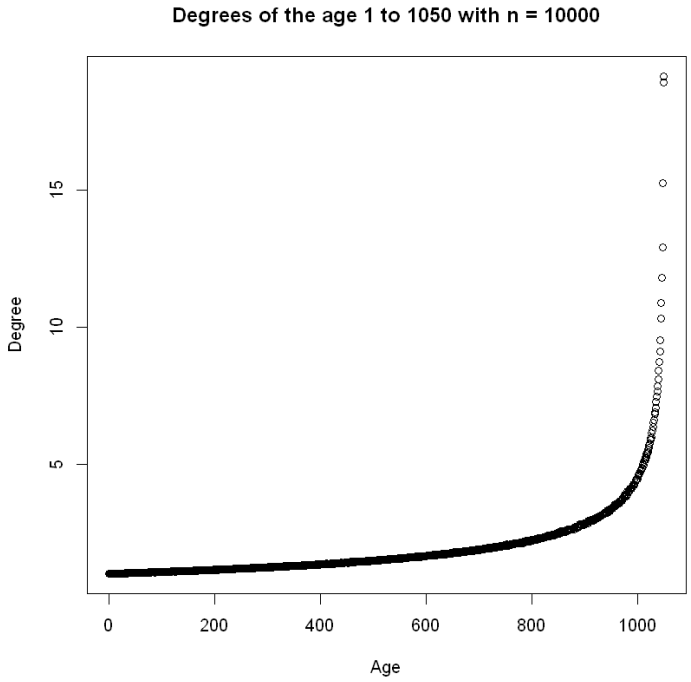
n=1050: Slope= -1.175



n=10500: slope= -1.49

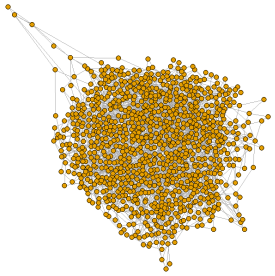
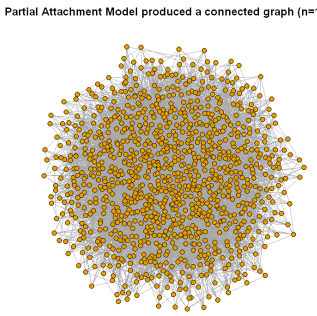
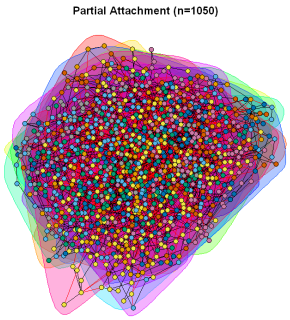
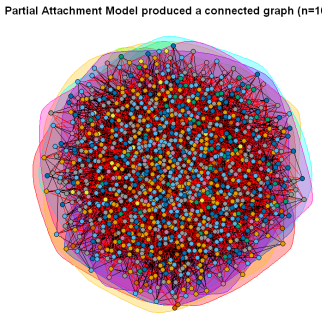
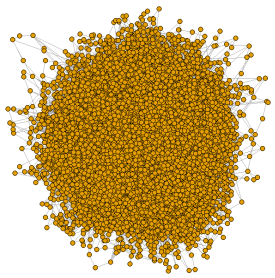
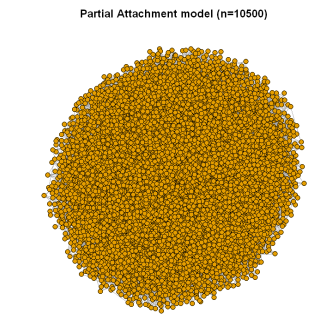


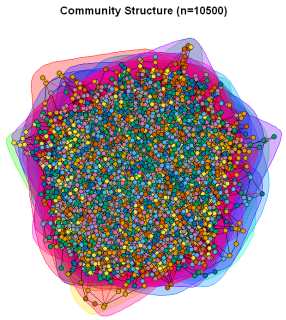
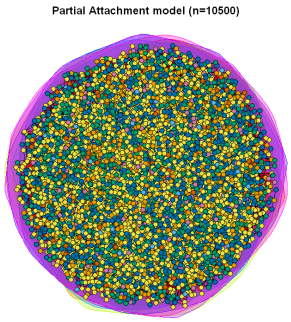
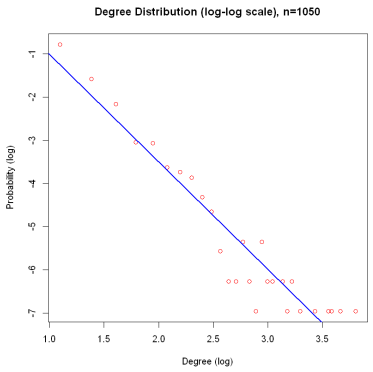
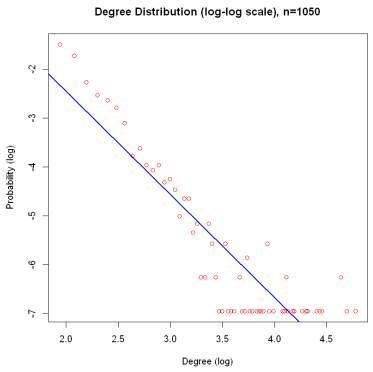
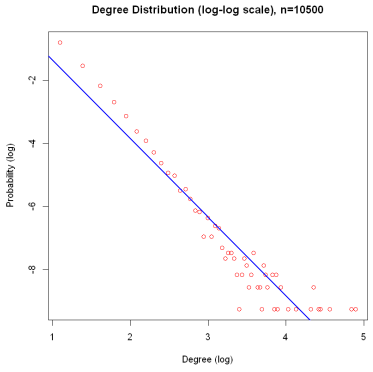
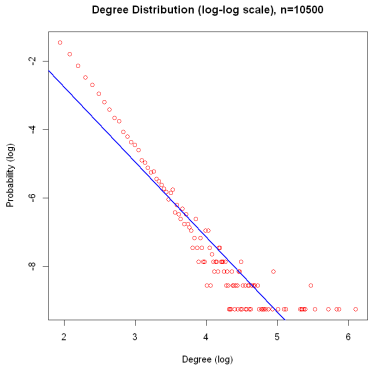
Part f

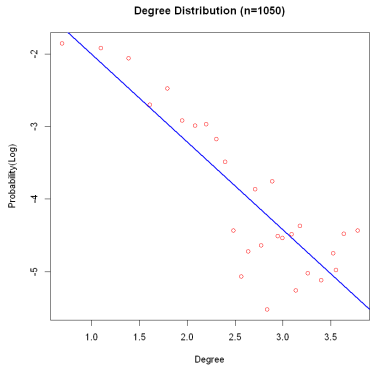
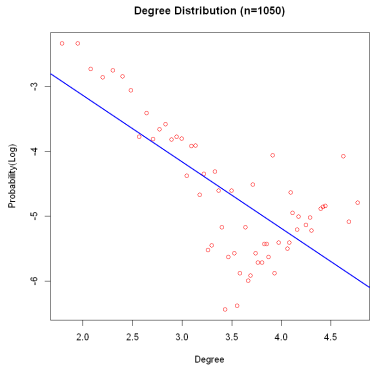
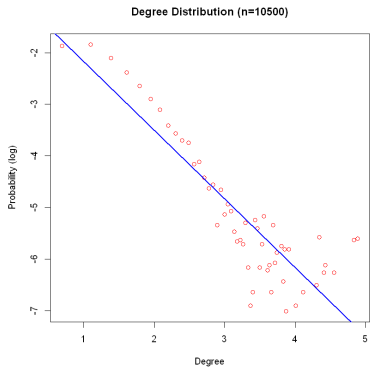
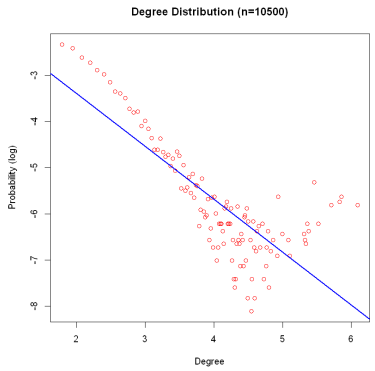
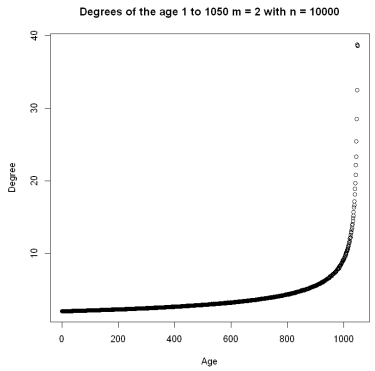
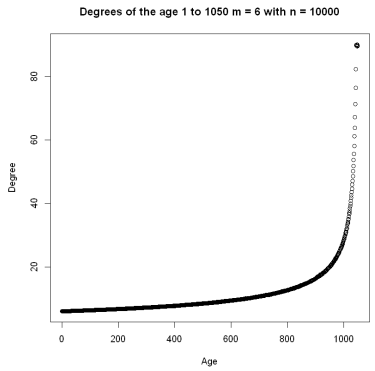


This confirms our intuition that the newest added node (the youngest) will have the lowest degree and the oldest nodes have the highest degree. This is due to the “rich getting richer” feature of preferential attachment. Nodes with higher degrees have a higher probability of gaining new links than those with less degrees. Obviously, newer nodes will have less degree than that of the older nodes.

Part g

	m=2	m=6
Undirected network n=1050		
Community structure		
Modularity	0.521962364	0.24488374
Assortativity	-0.0431344	-0.013944
Undirected network n=10500		

Community structure		
Modularity	0.5307466	0.24599312
Assortativity	-0.005583	-0.00016054
Degree distribution (n=1050)		
Slope	-2.486	-2.114
Degree distribution (n=10500)		
Slope	-2.490	-2.198

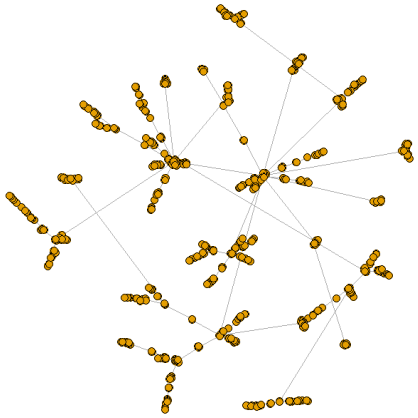
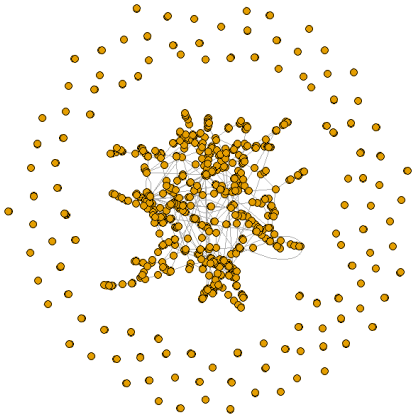
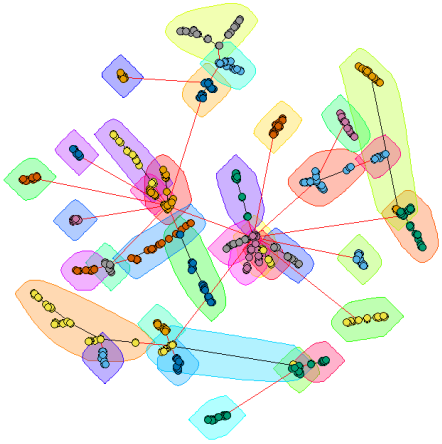
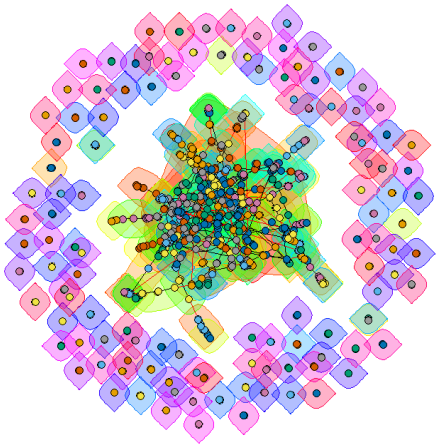
Randomly picked degree distribution (n=1050)		
Slope	-1.2097	-1.025
Randomly picked degree distribution (n=10500)		
Slope	-1.329	-1.146
Expected degree vs. Age (n=1050)		

The general trend is that with a larger m, the modularity decreases, which suggests that there are dense connections between nodes in clusters, which we see demonstrated by the community structure graph. This makes sense because a higher number of connections are made from the incoming node introduced to the graph, and these nodes are likely to make connections with degree rich nodes. Hence, the rich get richer and denser.

Additionally, the slope of the log-log scale of degree distribution stays linear and decreases for higher m. This is the same observation for the degree distribution for the random jth node. We see that while we increase m (the number of nodes that a new node attaches to), we stray from the expected behavior of a preferential attachment model where the power exponent is 3.

Finally, the relationship between degree and age of node is still valid where the older the node is, the higher the expected degree.

Part h

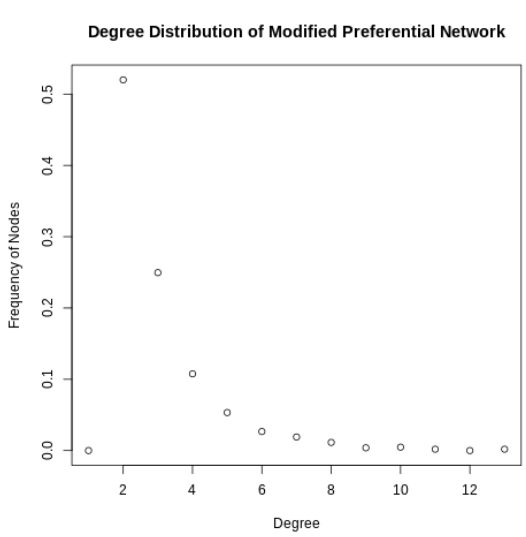
Original	Matched
	
	
Modularity of Original: 0.9241685531001878	Modularity of Matched: 0.8282298907398281
Assortativity: **-0.11491808**	Assortativity: -0.0212100644
Connected	Not connected

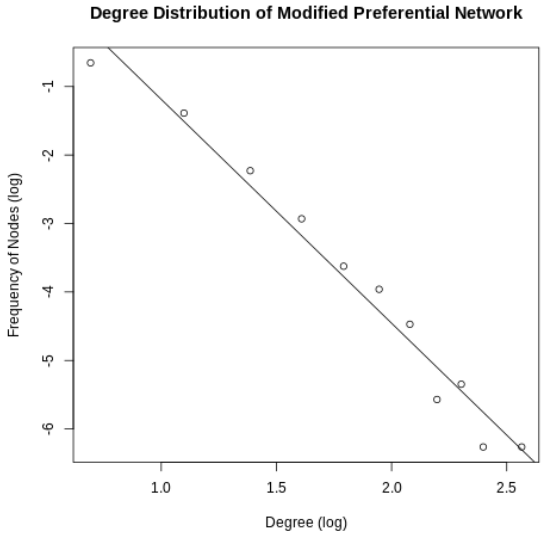
One clear difference between the two is that the original graph is fully connected, but the matched graph has a ring of isolated nodes that are not connected surrounding the GCC. Therefore, the lower modularity for the matched graph makes a lot more sense. Otherwise, these two graphs have the same degree distribution and the same average degree.

Even though these two graphs have the same degree distribution and same average degree, stub matching does not create the original power law distribution graph we initially expected. Although the modularity is similar, which means there are clusters (hubs), it did not result in the same connectedness.

Problem 3: Modified Preferential Attachment Model

Part a



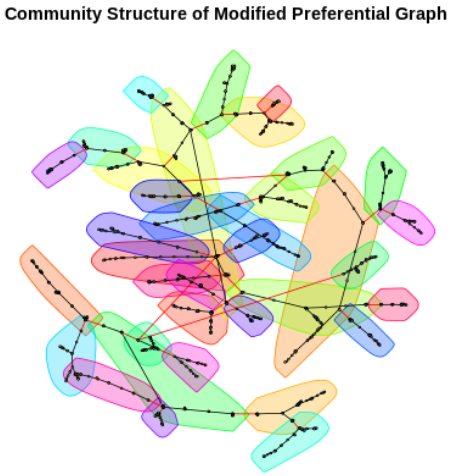


Power law degree distribution occurs where $P_k = \frac{c(k_{max}, \gamma)}{k^{\gamma}}$

At steady state, for a preferential attachment model, ideally, the degree distribution is a power law with power law exponent -3 as shown in class.

The power law exponent is k where $y = ax^k$. We can find the power law exponent by turning our degree distribution into a log-log plot and finding the slope of the line that fits the graph. The slope of the linear fit is -3.269, so the power law exponent is -3.269. The slope conforms very closely to what we expected to see from class when $k >> 1$, then $\lim_{k \to \infty} \alpha \frac{1}{k^3} = k^{-3}$. In theory, the power law exponent is -3 and our experimentally found power law exponent is close to that!

Part b



Modularity is 0.936335. In comparison with the original preferential graph, the modified preferential graph, which penalizes the age of a node, has denser connection between nodes within modules, which we can see from the higher modularity.

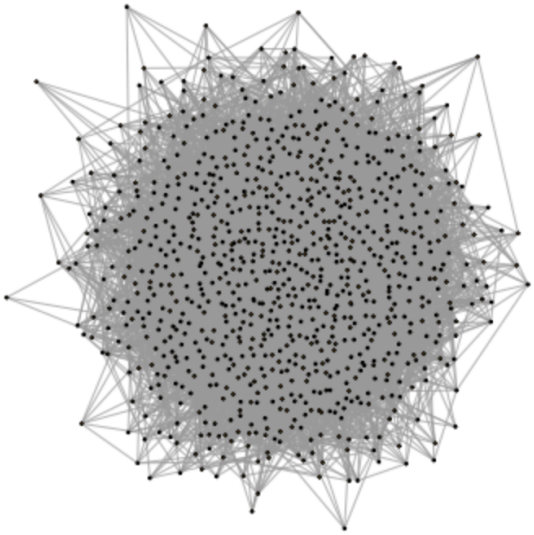
Part 2: Random Walk on Networks

Problem 1: Random Walk on Erdos-Renyi (ER) networks

Part a

This task called for the creation of an undirected random network with 900 nodes, with the probability of drawing an edge between any pair of nodes being .015, or 1.5%. This was accomplished with the use of the sample_gnp function, which generates random graphs according to the G(n,p) Erdos-Renyi model. Providing the number of vertices as 900 and the probability as .015, as well as indicating for it to be undirected, the following could be generated and plotted.

Figure Part 2 Problem 1 Part a: Network with n = 900 and p = .015

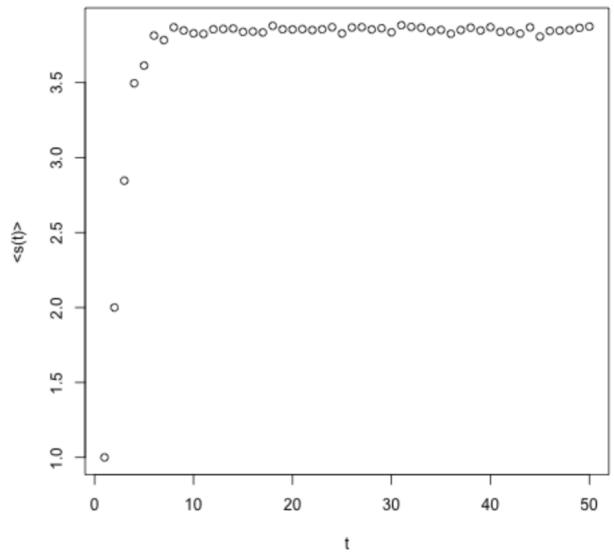


Part b

In this task, the aim was to measure the average distance (the distance was defined as the shortest path length) and the variance of that distance that a random walker would stray from its initial location.

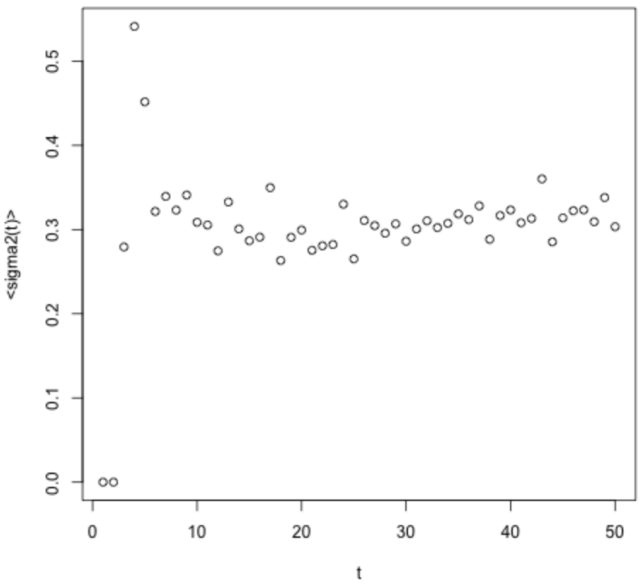
Random walks were performed from random initial starting points for different amounts of steps, in this case from 1 step to 50 steps. For each step choice, we ran 1000 iterations of the random walk, from which the average shortest path was measured. The following graph depicts the average distance versus the number of steps taken.

Figure Part 2 Problem 1 Part b: Average Distance vs Steps



We see that as the steps increase, the average distance increases towards a steady state of around 3.8. Similarly, we can extract the variance in the distances.

Figure Part 2 Problem 1 Part b: Distance Variance vs Steps



Here we see that as the step size increases, the variance in the shortest distance reaches a steady state of around 0.3 after initially seeing values in the range of 0.55. Lower step values cover less of the network

and are less able to average out the probabilistic fluctuations in path lengths. More steps can better discover the network and smooth out the variance inherent to the random process.

Part c

Here we measured the degree distribution of the nodes that were reached at the end of the random walks. This was done for 900 iterations and collected for steps equal to 50. The degrees were collected and plotted as seen below. The degree distribution for the graph was also produced.

Figure Part 2 Problem 1 Part c: Distribution of Degrees for End Nodes

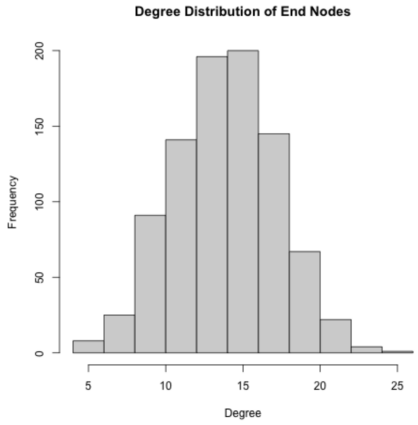
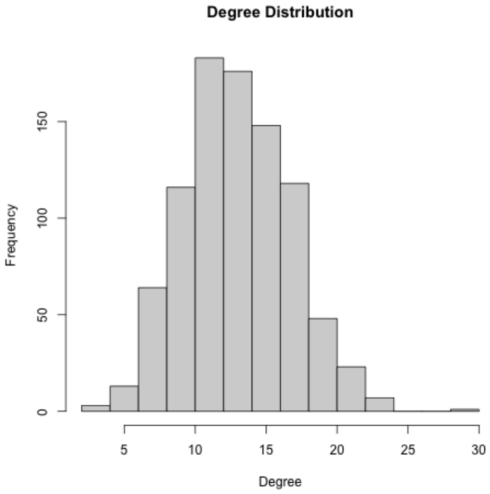


Figure Part 2 Problem 1 Part c: Distribution of Degrees for Graph



A comparison of the two shows a similar binomial distribution expected for the random network. The task essentially performs a random sampling so we would naturally expect to see this. Of course some variance is also to be expected.

Part d

Here we repeat the calculations we performed in part b at a larger scale, increasing the nodes of the network to 9000.

Figure Part 2 Problem 1 Part d: Average Distance vs Steps

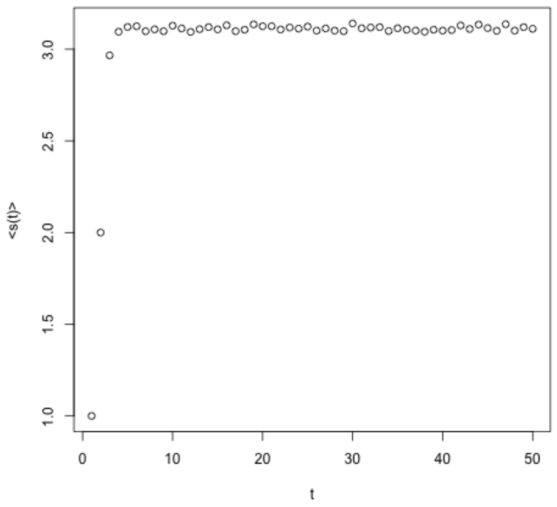


Figure Part 2 Problem 1 Part d: Distance Variance vs Steps

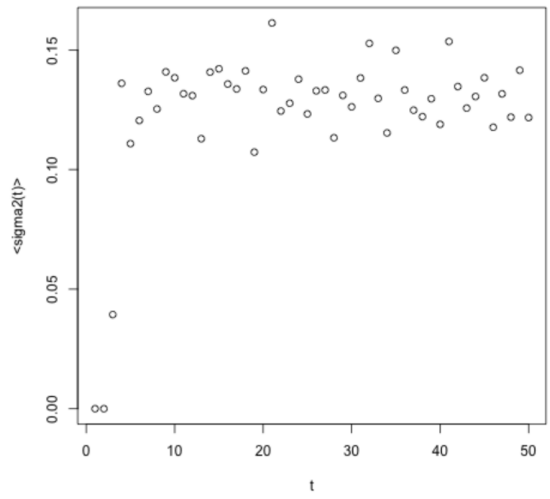
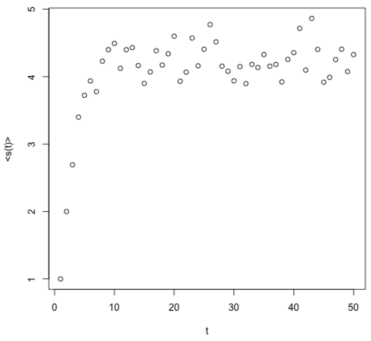
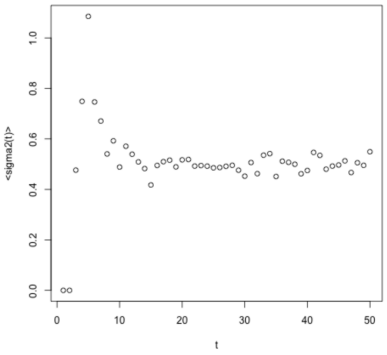
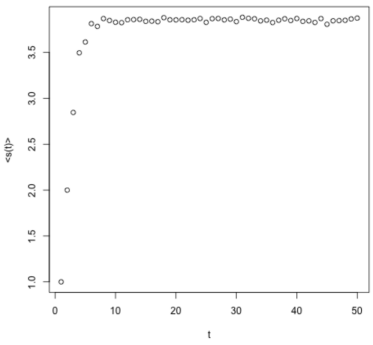
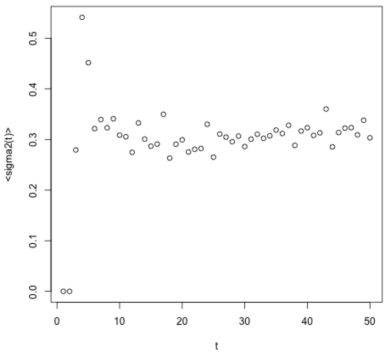
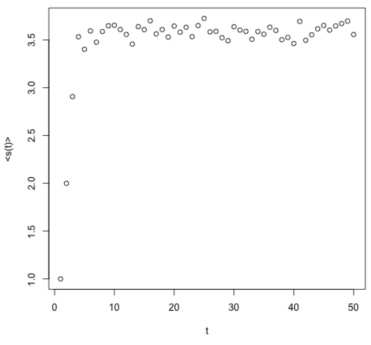
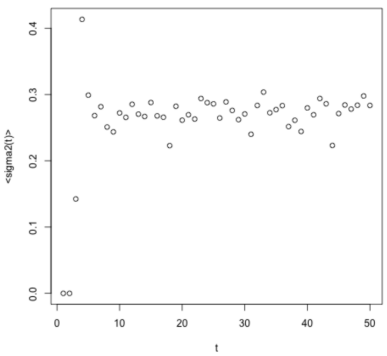
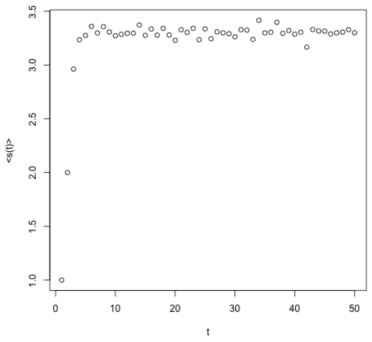
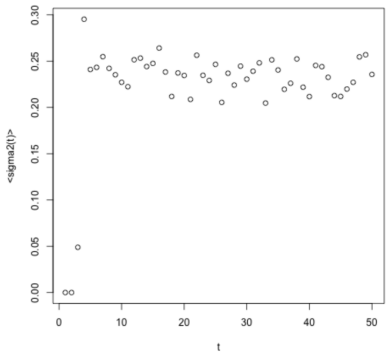
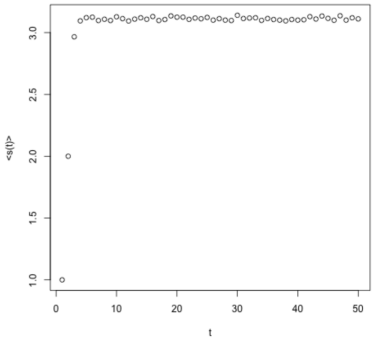
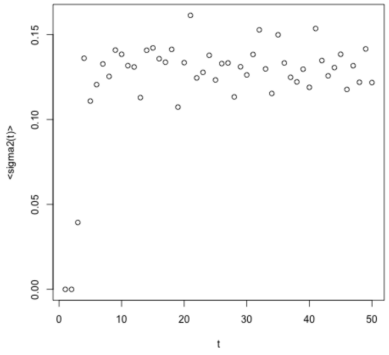


Table Part 2 Problem 1 Part d: Diameter and Average Distance for Various Network Sizes

Nodes	Diameter	<s(t)>	Variance
500	6		
900			
2000	4		

5000	3		
9000	3		

In comparing the average shortest distance paths, we see a smaller average distance for the increasingly larger networks. This can be attributed to the fact that a larger network comes with greater connectivity; there are more paths to take. Additionally, larger networks have smaller diameters. The diameter of a network is the shortest distance between the two most distant nodes in a network, hence a smaller diameter would mean shorter average paths between nodes. And since the network is more concentrated, we naturally see the variance also decrease. Moreover, we notice that the larger node networks converge to their steady state average distance and variance values in less time steps than the smaller node networks. This can also be attributed to the diameter size. Since the larger node networks have demonstrated a smaller diameter, the graph is more condensed and takes a shorter amount of time steps to converge to steady state values.

Problem 2: Random walk on networks with fat-tailed degree distributions

Part a

In this section we work with a network generated according to a fat-tailed degree distribution, that being one where there are few nodes with high degrees (also called hubs) that are connected to many other nodes, with the majority of nodes having lower degrees.

We generate a preferential attachment network with 900 nodes where each node is connected to m = 1 old nodes. This is done via igraph's sample_pa function. The following is produced.

Figure Part 2 Problem 2 Part a: P.A. Network with n = 900 and m = 1

We see the expected clustering around "hub" nodes.

Part b

We perform a similar analysis as was done in part b of the first problem. Steps were increased to get a better picture, and the iterations were lower due to processing capacity. The results are below.

Figure Part 2 Problem 2 Part b: Average Distance vs Steps for P.A. Network

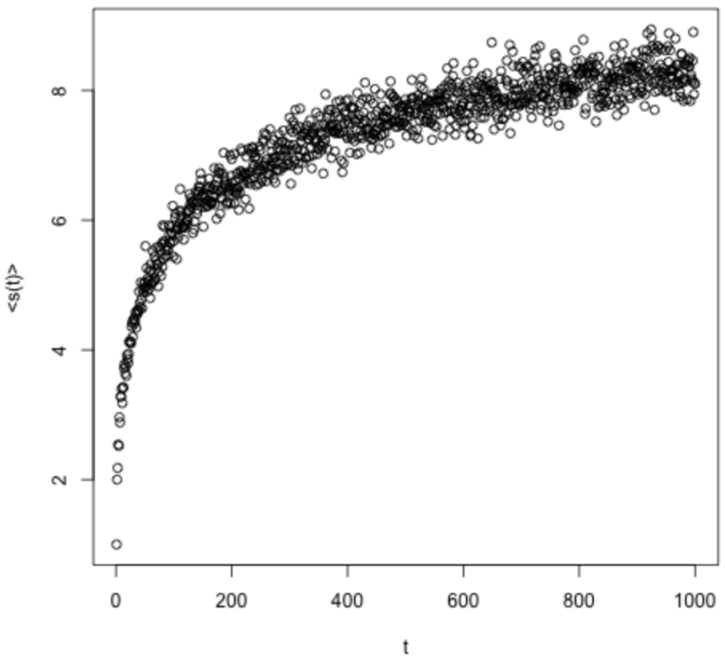
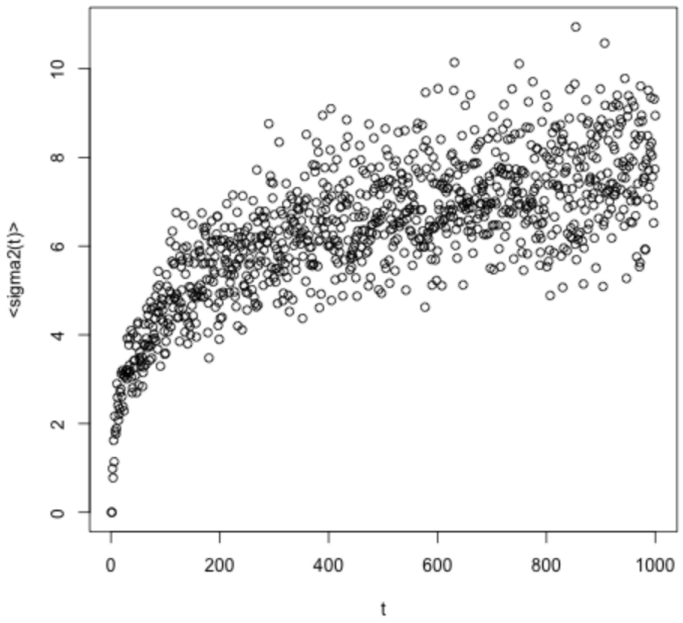


Figure Part 2 Problem 2 Part b: Distance Variance vs Steps for P.A. Network



Looking at the average distance we see the value asymptote at slightly over 8, a marked jump from what we saw earlier in our non-preferential attachment model. This seems to indicate that the random walker may find itself getting “stuck” in a cluster of poorly attached nodes; we can see these little chains in the

depiction from part a. These areas of poor connectivity affect the average shortest path and introduce a great deal of variance to the results, because in tandem there are areas of good connectivity, namely around the hubs. Moreover, the fat-tailed preferential attachment did not reach its steady state value as quickly as the ER network. It seemed to reach steady state in around 250 time steps.

Part c

Once again we repeat our analyses from problem 1. The degree distribution for the end nodes and graph is shown below.

Figure Part 2 Problem 2 Part c: Distribution of Degrees for End Nodes for P.A. Graph

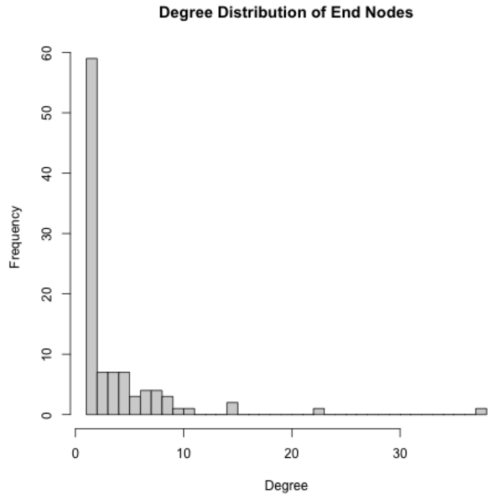
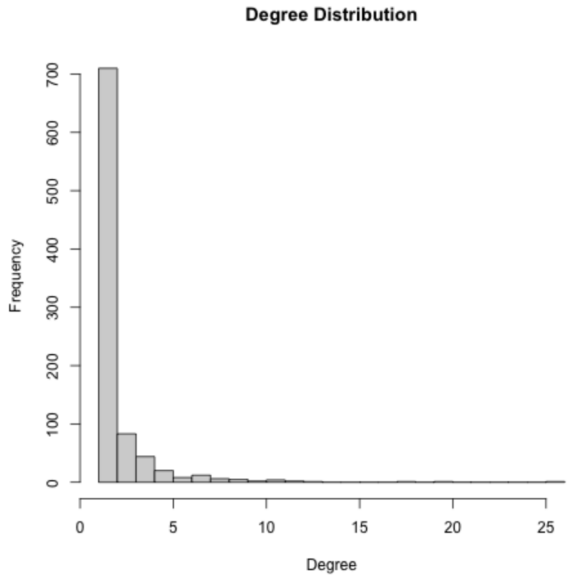


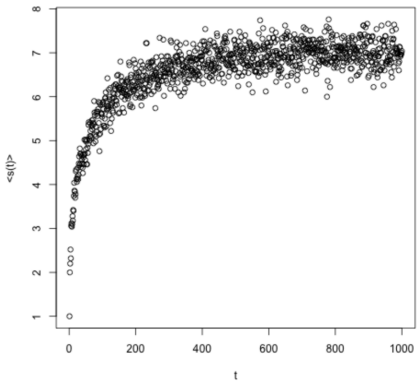
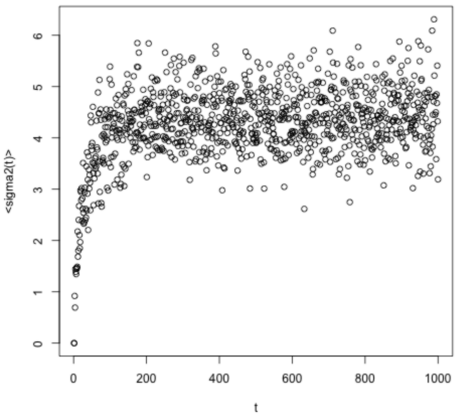
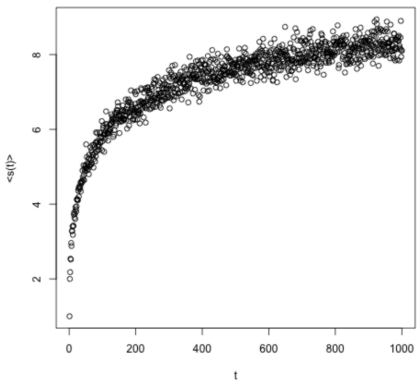
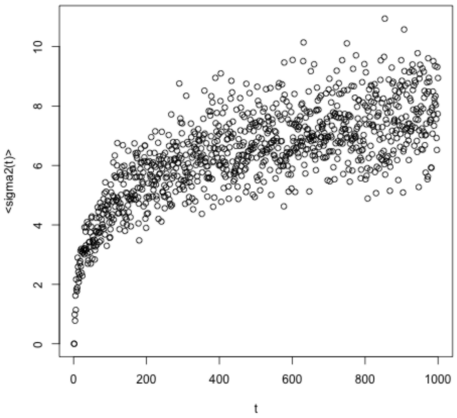
Figure Part 2 Problem 2 Part c: Distribution of Degrees for P.A. Graph

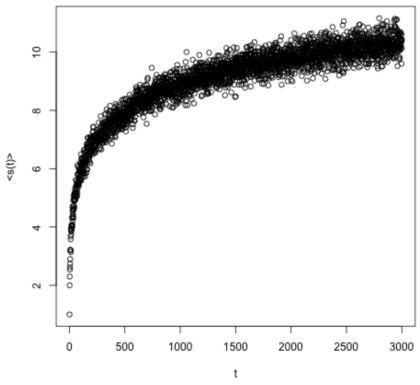
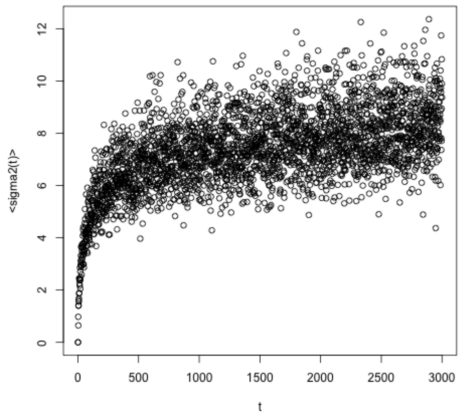


The preferential attachment model creates a network where there are few nodes with high degrees that are connected to many other nodes, with the majority of nodes having lower degrees. We see this clearly in our degree distributions, with lower degrees taking up a majority of the distribution, with a very small number of highly connected nodes. The distributions agree. Since the preferential attachment network we modeled had a fat-tailed distribution, we expect that the model has power-law degree distribution, which we observe in the experiment. While we initially observed a binomial distribution for the ER graph, we observed a quickly decay degree distribution for the Preferential attachment with fat-tail distribution.

Part d

We now do the analysis of part b, varying the number of nodes to 90 and 9000, maintaining m = 1. The table below provides the results, along with the diameter of the graphs.

Nodes	Diameter	<s(t)>	Variance
90	11		
900	23		

9000	31		

In contrast to the simple undirected network from problem 1, we see that for the preferential attachment network, the diameter increases with increasing network size. This can be explained with similar reasoning from part 2. Although there are hubs with high levels of connectivity, nodes with weak connectivity outnumber them significantly, as we saw in the degree distribution. As the network grows, the rich (the hubs) get richer and more connected and the poor get poorer (less connected). The increase in average shortest path length and its variance, for reasons discussed in part 2, also go along with an expected increase in diameter of the network as some nodes become more isolated.

Problem 3: PageRank

The PageRank algorithm is an algorithm used by Google Search to rank websites in their search engine results.The algorithm works by analyzing the links between web pages, assigning each page a score based on the number and quality of links that point to it. Pages with a higher PageRank score are considered more important and are more likely to appear at the top of search engine results. The PageRank algorithm is based on the principle of "voting" or "recommendation". A web page that is linked to by many other pages is considered to be more valuable or important than a page that has few or no links pointing to it. The algorithm calculates a score for each page based on the number and quality of incoming links, as well as the PageRank score of the pages that are linking to it. The more links a page has from high-quality sources, the higher its PageRank score will be.

Part a

The described network is constructed as follows:

1) First two PA networks are created with the given n = 1000 and n = 4 values.
2) Then the nodes of second network is permuted with the permute() method
3) Edges of the permuted network are added to the first graph created.

In result, the combined network consists of 7180 edges.

In order to calculate the probability that the walker visits each node. Random walk on this network is simulated 100 times with each walk consisting of 1000 steps. Start of each node is selected randomly from the graph. Then, each visit to a node is summed across all random walks and then normalized by total visits. The steps where the random walk did not reach steady state are discarded from the sum. We have approximated the convergence point as log(n) where n is the number of steps required for convergence.

Probability of visiting node j given that the current walker is at node i for Problem 3 Part a can be written as follows:

$$P_{ij} = \frac{1}{k_{out}(i)} A_{ij}$$

Therefore total probability of visiting node j after convergence can be written as follows:

$$\pi(j) = \sum_{i=1}^{n} P_{ij} \pi(i)$$

$$\pi(j) = \sum_{i=1}^{n} \frac{1}{k_{out}(i)} A_{ij} \pi(i)$$

A_{ij} is the adjacency matrix of a graph, in our case the graph is the combined graph above. n is the number of nodes in the graph. Note that this equation is only valid for Problem 3.a, where there is no teleportation probability.

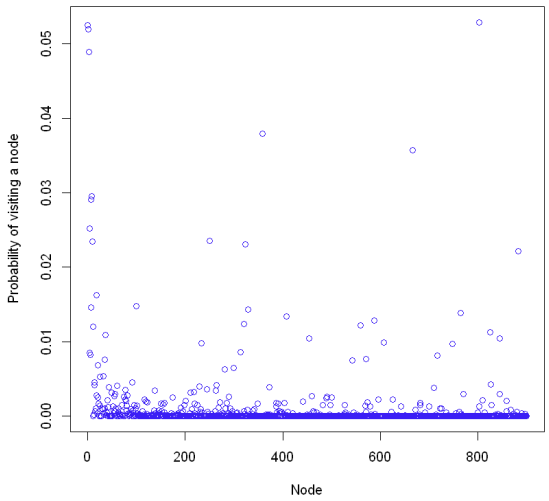


Figure Part 2 Problem 3.a.1: Probability that the walker visits a node

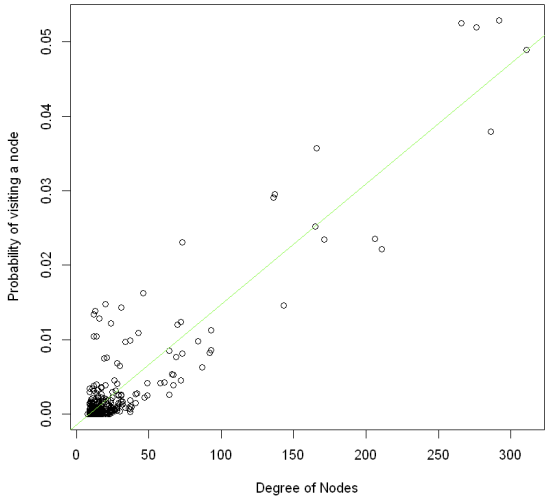


Figure Part 2 Problem 3.a.2: Linear Relationship between probability of visiting and degree of the nodes

As it can be seen from the above figure, there is a linear relationship between probability of visiting a node and degree of that node. This makes sense, since it is more probable to visit a node that has more edges. Note that, there is no teleportation probability, therefore it is not possible for random walkers to jump nodes that are not adjacent to the current node. Pearson correlation between probability of node and degree of that node is **0.9213344**. Fit parameters are given as below.

Fit parameters	Value
Intercept	-0.001474

Slope	**0.000162**

Table Part 2 Problem 3.a.1: Fit parameters for the fit in figure Part 2 Problem 3.a.2.

Part b

In this part, it is also possible to teleport to a random node with a probability of α = 0.2 while randomly walking. Other than teleportation, the simulation procedure is the same as the simulation procedure in part a. Therefore, a random walker will land on an adjacent node with probability 1- α = 0.8 and will land on a random node with probability α = 0.2.

Probability of visiting node j given that the current walker is at node i for Problem 3 Part b can be written as follows:

$$P_{ij} = (1 - \alpha)\frac{1}{k_{out}(i)}A_{ij} + \alpha\frac{1}{n}$$

Therefore total probability of visiting node j after convergence can be written as follows:

$$\pi(j) = \sum_{i=1}^{n} P_{ij}\pi(i)$$

$$\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i) + \alpha\frac{1}{n}$$

A_{ij} is the adjacency matrix of a graph. Note that this equation is only valid for Problem 3 Part b, where each node has equal teleportation probability. Also, teleportation probability of random walker is chosen as α = 0.2. n is the number of nodes in the graph. This equation is also referred to as the standard PageRank algorithm.

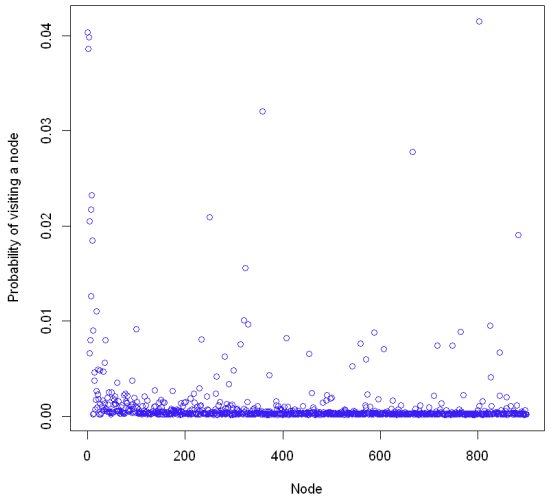


Figure Part 2 Problem 3.b.1: Probability that the walker visits a node with teleportation probability 0.2

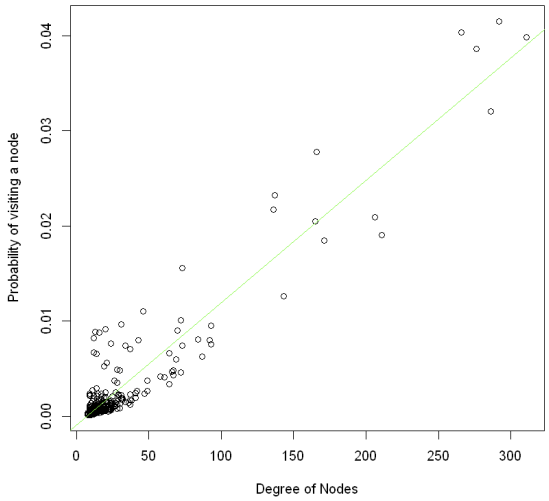


Figure Part 2 Problem 3.b.2: Linear Relationship between probability of visiting and degree of the nodes with teleportation probability 0.2

As it can be seen from the above figure, there is a linear relationship between probability of visiting a node and degree of that node. This makes sense, since it is more probable to visit a node that has more edges. Note that, there is teleportation probability, therefore it is possible for random walkers to jump nodes that are not adjacent to the current node. Slope is less than part a which makes sense since it is possible to teleport to nodes with less number of edges in this part. Number of visits to nodes with a high number of edges is smaller in this part. Pearson correlation between probability of node and degree of that node is **0.9491821.** Fit parameters are given as below.

Fit parameters	Value
Intercept	-0.001474
Slope	**0.0001286**

Table Part 2 Problem 3.b.1: Fit parameters for the fit in figure Part 2 Problem 3.b.2.

Problem 4: Personalized PageRank

Personalized PageRank is a variant of the PageRank algorithm that takes into account the interests and preferences of a particular user. Instead of using the same PageRank scores for all users, Personalized PageRank calculates a customized score for each user based on their specific interests and browsing history.

In traditional PageRank, the importance of a web page is based on the number and quality of links pointing to it. However, Personalized PageRank adds an additional layer of personalization by taking into account a user's past behavior and preferences.

Part a

The simulation in this part is very similar to the simulation in Part 2 Problem 3.b and the result is similar to the simulation in Part 2 Problem 3.a. In the previous part, we have assumed that it is equally probable to teleport to any node. However, this is unlikely in real world situations. Some webpages are more important than others and someone needs to define the importance of a webpage.

In this part, these importances are described by the page rank algorithm.The algorithm calculates a score for each page based on the number and quality of incoming links, as well as the PageRank score of the pages that are linking to it. The more links a page has from high-quality sources, the higher its PageRank score will be. We can use these page rank scores to teleport to other nodes, since we rely upon Google to decide which website to teleport.

We have used the page_rank function to obtain teleportation probabilities of each node. Using these probabilities and transition probabilities, several random walkers are simulated to obtain node visitation probabilities. Teleportation probability is chosen as 0.2 as given in the assignment.

Probability of visiting node j given that the current walker is at node i for Problem 4 Part a can be written as follows:

$$P_{ij} = (1 - \alpha)\frac{1}{k_{out}(i)}A_{ij} + \alpha\pi(j)$$

Therefore total probability of visiting node j after convergence can be written as follows (assuming page rank has converged):

$$\pi(j) = \sum_{i=1}^{n} P_{ij}\pi(i)$$

$$\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i) + \alpha\pi(j)$$

$$\pi(j) - \alpha\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i)$$

$$\pi(j) - \alpha\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i)$$

$$\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i) + \alpha\pi(j) = \sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i)$$

A_{ij} is the adjacency matrix of a graph. Note that this equation is only valid for part Problem 4 Part a since our teleportation probability relies on the page rank score of that page. **In theory, the result is the same as 3.a.** Therefore, it is redundant to add teleportation term to the equation.

Also, teleportation probability of random walker is chosen as $\alpha = 0.2$. n is the number of nodes in the graph.

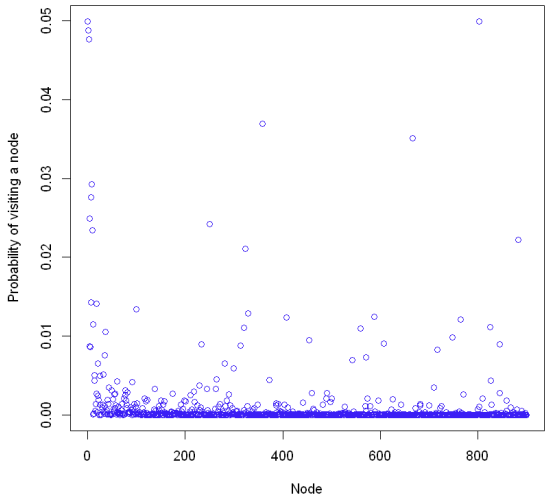


Figure Part 2 Problem 4.a.1: Probability that the walker visits a node with teleportation probability 0.2. Teleportation probability of each node is found with the PageRank algorithm.

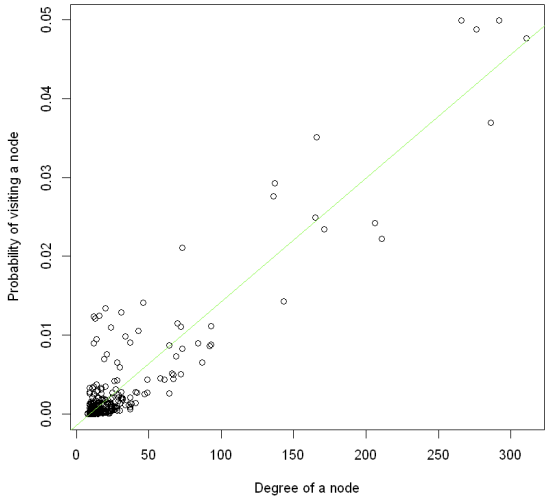


Figure Part 2 Problem 4.a.2: Linear Relationship between probability of visiting a node and degree of the nodes with teleportation probability 0.2. Teleportation probability of each node is found with the PageRank algorithm.

As it can be seen from the above figure, there is a linear relationship between probability of visiting a node and degree of that node. This makes sense, since it is more probable to visit a node that has more edges. Note that, there is teleportation probability, therefore it is possible for random walkers to jump nodes that are not adjacent to the current node.

Slope is very similar to slope in part 3.a. This makes sense since we can only teleport to the nodes that have high page rank, therefore it is highly not probable to teleport to the nodes with low page rank. **As we have shown above, using page rank probabilities as teleportation probabilities of each node is same**

as not using teleportation in the model. Practical simulation validates this result. This part can be also interpreted as trapped at high degree nodes.

The small difference between slopes can be explained by the default page rank algorithm and our random visitor. We always use converged page rank probability for teleportation, therefore our algorithm might have converged to a different page rank than the part 3.a. Pearson correlation between probability of node and degree of that node is **0.9306456.** Fit parameters are given as below.

Fit parameters	Value
Intercept	-0.0013877
Slope	**0.0001566**

Table Part 2 Problem 4.a.1: Fit parameters for the fit in figure Part 2 Problem 4.a.2.

Part b

In this part, rather than using page rank probabilities, we have found the 2 nodes that correspond to the median of the page rank probabilities. We have set the teleportation probability to that nodes as 0.5. Other nodes have 0 teleportation probability.

Probability of visiting node j given that the current walker is at node i for Problem 4 Part b can be written as follows:

$$P_{ij} = (1 - \alpha)\frac{1}{k_{out}(i)}A_{ij} + \alpha\, B(j)$$

which can be written as a piecewise function. We need to define a subset of median nodes which is denoted by T and assume that there is equal probability to teleport to nodes in the T. Note that for this part, T only consists of the **median nodes of page rank order**.

Then, above equation transforms to the following:

$$P_{ij} = (1 - \alpha)\frac{1}{k_{out}(i)}A_{ij} + \alpha\frac{1}{|T|} \quad j \in T$$

$$P_{ij} = (1 - \alpha)\frac{1}{k_{out}(i)}A_{ij} \quad j \notin T$$

Therefore total probability of visiting node j after convergence can be written as follows:

$$\pi(j) = \sum_{i=1}^{n} P_{ij}\pi(i)$$

$$\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i) + \alpha\frac{1}{|T|} \quad j \in T$$

$$\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i) \quad j \notin T$$

A_{ij} is the adjacency matrix of a graph. Note that this equation is only valid for part Problem 4 Part b. Also, teleportation probability of random walker is chosen as $\alpha = 0.2$. n is the number of nodes in the graph.

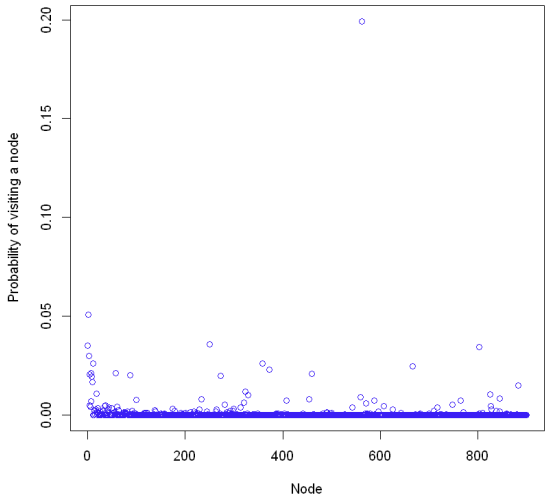


Figure Part 2 Problem 4.b.1: Probability that the walker visits a node with teleportation probability 0.2. Teleportation probability of median nodes are 0.5, others have 0.

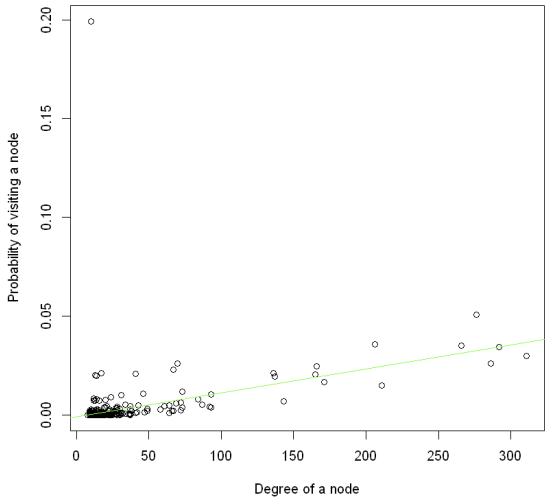


Figure Part 2 Problem 4.b.2: Linear Relationship between probability of visiting a node and degree of the nodes with teleportation probability 0.2. Teleportation probability of page rank median nodes is 0.5, other nodes have 0.

As it can be seen from the above figure, there is a linear relationship between probability of visiting a node and degree of that node. This makes sense, since it is more probable to visit a node that has more edges. Note that, there is teleportation probability, therefore it is possible for random walkers to only teleport to the median nodes that are not adjacent to the current node. Since teleportation to these nodes can occur, these 2 nodes have nearly 0.2 visitation probability. Other nodes have similar probability with 3.a, since there is no teleportation for those nodes. Pearson correlation between probability of node and

degree of that node is **0.4280885.** Note that this correlation has reduced due to the two nodes. Fit parameters are given as below.

Fit parameters	Value
Intercept	-0.0008238
Slope	**0.0001213**

Table Part 2 Problem 4.b.1: Fit parameters for the fit in figure Part 2 Problem 4.b.2.

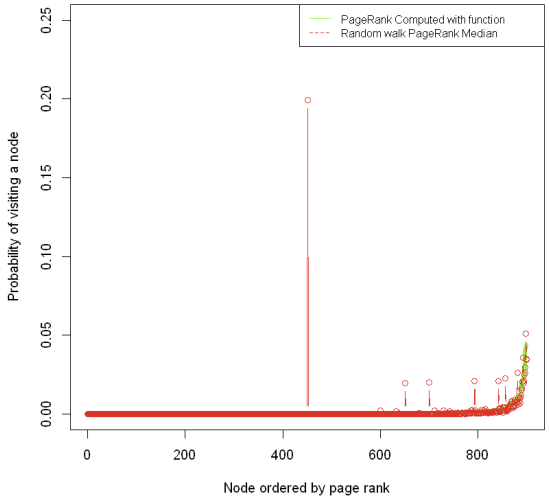


Figure Part 2 Problem 4.b.3: Probability that the walker visits a node with teleportation probability 0.2. Teleportation probability of median nodes are 0.5, others have 0. Nodes are ordered by page rank. Therefore, Node 1 has the lowest probability.

As it can be seen from the above graph, there are two spikes at the median of the nodes. These nodes have a high probability since they have a teleportation probability of 0.5. Also, note that there are small spikes at the right side of spikes which are the adjacent nodes of median nodes. This makes sense because a random walker could proceed to the adjacent nodes after teleporting to the median node. The reason why they appear at the right side of spikes is that those nodes are not only adjacent to the median nodes, they also have a high number of edges. Therefore, it is also possible that they could have been visited at the third or more visit after teleportation.

Part c

Standard page rank equation uses the following scheme to model web surfing:

$$P_{ij} = (1 - \alpha)\frac{1}{k_{out}(i)}A_{ij} + \alpha\frac{1}{n}$$

Therefore total probability of visiting node j after convergence can be written as follows:

$$\pi(j) = \sum_{i=1}^{n} P_{ij}\pi(i)$$

$$\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i) + \alpha\frac{1}{n}$$

However, the standard equation assumes that people's interest on all sites are equally weighted which is not true in the real world. One needs to add a self-reinforcement factor into the page rank equation to model an actual surfer.

We can consider the case when the surfer teleports to a set T of trusted nodes with equal probability and does not teleport to nodes that are not trusted.

$$P_{ij} = (1 - \alpha)\frac{1}{k_{out}(i)}A_{ij} + \alpha\, B(j)$$

$$B(j) = \frac{1}{|T|} \text{ if } j \in T$$

$$B(j) = 0 \text{ if } j \notin T$$

which can be written as a piecewise function.

$$P_{ij} = (1 - \alpha)\frac{1}{k_{out}(i)}A_{ij} + \alpha\frac{1}{|T|} \quad j \in T$$

$$P_{ij} = (1 - \alpha)\frac{1}{k_{out}(i)}A_{ij} \quad j \notin T$$

Therefore total probability of visiting node j or page rank of j after convergence can be written as follows:

$$\pi(j) = \sum_{i=1}^{n} P_{ij}\pi(i)$$

$$\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i) + \alpha\frac{1}{|T|} \quad j \in T$$

$$\pi(j) = (1 - \alpha)\sum_{i=1}^{n} \frac{1}{k_{out}(i)}A_{ij}\pi(i) \quad j \notin T$$

In result, random walkers will be trapped at high page rank nodes and trusted nodes.

For the case of Problem 4 Part b, $\frac{1}{|T|} = 0.5$ and T = {median1,median2}. The random walker was trapped at those two nodes, therefore the algorithm yielded a high probability for those nodes.