ECE 232E Project 1

Random Graphs and Random Walks

Chenchen Kuai	206074833
Hao Wang	405629183
Yuning Yang	705930008

1. Generating random networks

1. Random walk on Erdos-Renyi networks

(a) Create undirected random networks with n = 900 nodes, and the probability p for drawing an edge between two arbitrary vertices 0.002, 0.006, 0.012, 0.045, and 0.1. Plot the degree distributions. What distribution (linear/exponential/gaussian/binomial or something else) is observed? Explain why. Also, report the mean and variance of the degree distributions and compare them to the theoretical values.

Answer:

Degree distributions:

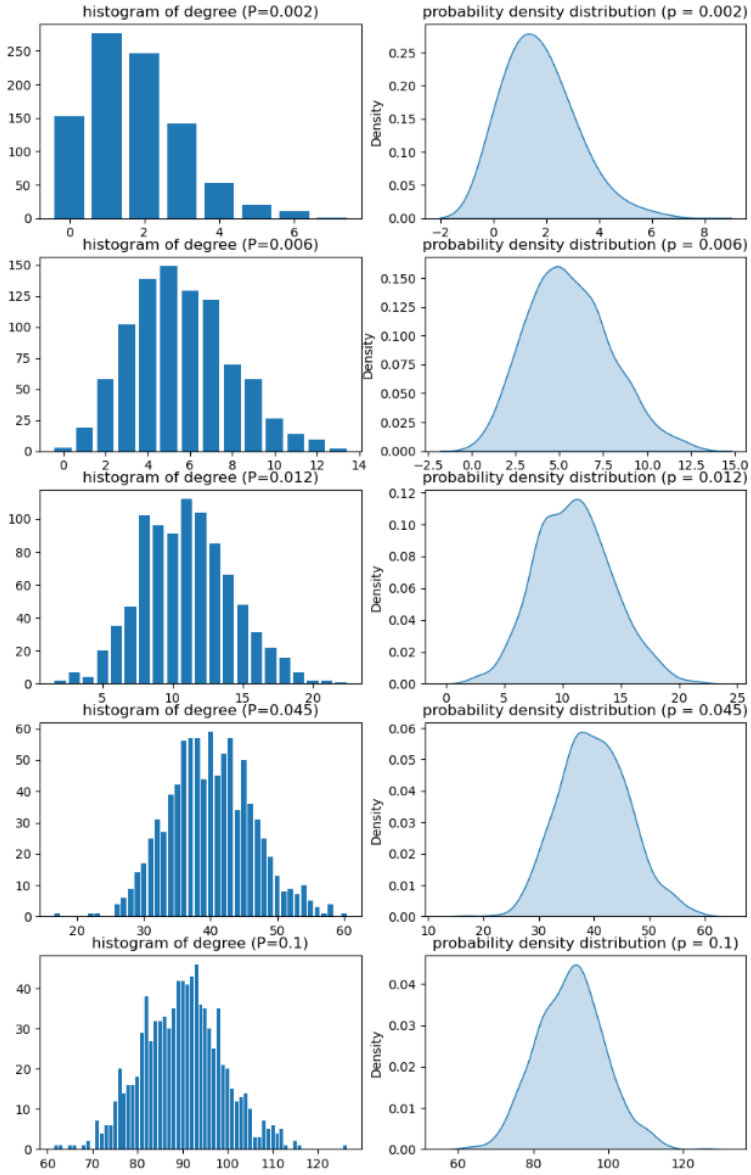


Figure 1. Distribution plot and histogram of degree (p = 0.002, 0.006, 0.012, 0.045,0.1)

A **binomial distribution** is observed. For a node, having an edge over the other node has the probability p, and not having an edge has the probability (1-p), and the probability is independent of the other edges' choices:

$$P(degree(v) = k) = C_{n-1}^{k} p^{k} (1 - p)^{n-1-k} \quad (1)$$

Thus, the distribution follows the binomial distribution:

$$degree \sim B(n - 1, p) \quad (2)$$

Moreover, when p is extremely small and n is extremely large, the binomial distribution is approximate to a poisson distribution:

$$P(degree(v) = k) = \frac{(np)^{k} e^{-np}}{k!} \quad (3)$$

Mean and variance of degree distributions:

p = 0.002, N = 900, mean +- sd: 1.7444 +- 1.3051, var = 1.703

p = 0.006, N = 900, mean +- sd: 5.4689 +- 2.3366, var = 5.460

p = 0.012, N = 900, mean +- sd: 10.8533 +- 3.3645, var = 11.320

p = 0.045, N = 900, mean +- sd: 40.1378 +- 6.3158, var = 39.890

p = 0.1, N = 900, mean +- sd: 89.6600 +- 8.9025, var = 79.255

Theoretical values:

$$E(x) = (n - 1)p \quad (4)$$

$$var(x) = (n - 1)p(1 - p) \quad (5)$$

Table 1. Mean and variance of the degree distribution

p	E(x)	Var(x)	Mean	Var	p
0.002	1.798	1.794	1.744	1.703	0.002
0.006	5.394	5.361	5.469	5.460	0.006
0.012	10.788	10.659	10.853	11.320	0.012
0.045	40.455	38.635	40.138	39.890	0.045

The mean and variance of the generated graph are in agreement with the theoretical values, with only small differences.

(b) For each p and n = 900, answer the following questions: Are all random realizations of the ER network connected? Numerically estimate the probability that a generated network is connected. For one instance of the networks with that p, find the giant connected component (GCC) if not connected. What is the diameter of the GCC?

Answer:

To estimate the probability of the ER network being connected, we apply 1000 runs and get the probability of these running results being connected. The results are shown in Table 2.

Table 2. GCC information of ER graphs generated

p	Probability of being connected	Is_connected	Diameter of GCC	Number of nodes in GCC	Number of edges in GCC
0.002	0	False	23	636	749
0.006	0.017	False	8	896	2480
0.012	0.985	True	5	900	4728
0.045	1	True	3	900	18186
0.1	1	True	3	900	40495

As p gets larger than 0.012, the network's connection probability will get close to 1. While p is smaller than 0.06, the probability of the network being connected is close to 0. The diameter of the GCC is becoming smaller as p becomes larger, and as the network become connected, the GCC will eventually be the size of the whole generated network.

(c) *It turns out that the normalized GCC size (i.e., the size of the GCC as a fraction of the total network size) is a highly nonlinear function of p, with interesting properties occurring for values where p = O(1/n) and p = O(lnn/n).*

For n = 900, sweep over values of p from 0 to a pmax that makes the network almost surely connected and create 100 random networks for each p. pmax should be roughly determined by yourself. Then scatter plot the normalized GCC sizes vs p. Plot a line of the average normalized GCC sizes for each p along with the scatter plot.

Answer:

P_{max} is defined 0.01, where the network is almost surely connected. Thus, the x-axis should be from 0 to 0.01.

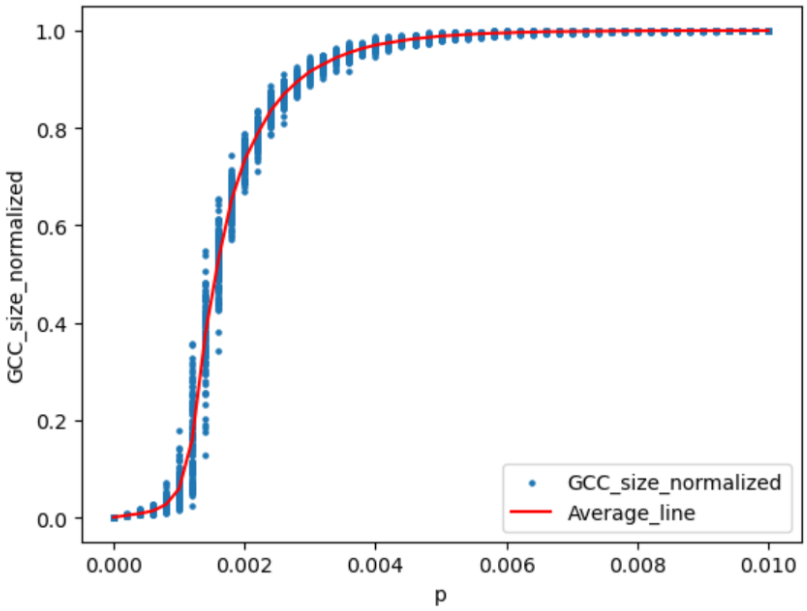


Figure 2. The scattered plot of normalized GCC sizes

i. Empirically estimate the value of p where a giant connected component starts to emerge (define your criterion of "emergence")? Do they match with theoretical values mentioned or derived in lectures?

Answer:

Defined emerge: After this point, when p becomes higher, the normalized size of GCC is emerging with a high slope. While before the point, the slope is relatively low. In other words, it is the point at which the network undergoes a transition from a disconnected state to a connected state.

From Figure 2, we can find that before 0.001, the slope is small, and after 0.001, the slope becomes big. The value 0.001 is close to O(1/n), which is 0.0011.

ii. Empirically estimate the value of p where the giant connected component takes up over 99% of the nodes in almost every experiment.

From Figure 2, we can find that when p is larger than 0.0075, 99% of the nodes in almost every experiment is connected. This value is in agreement with the theoretical value O(lnn/n), which is 0.0076.

1(d)

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

Answer:

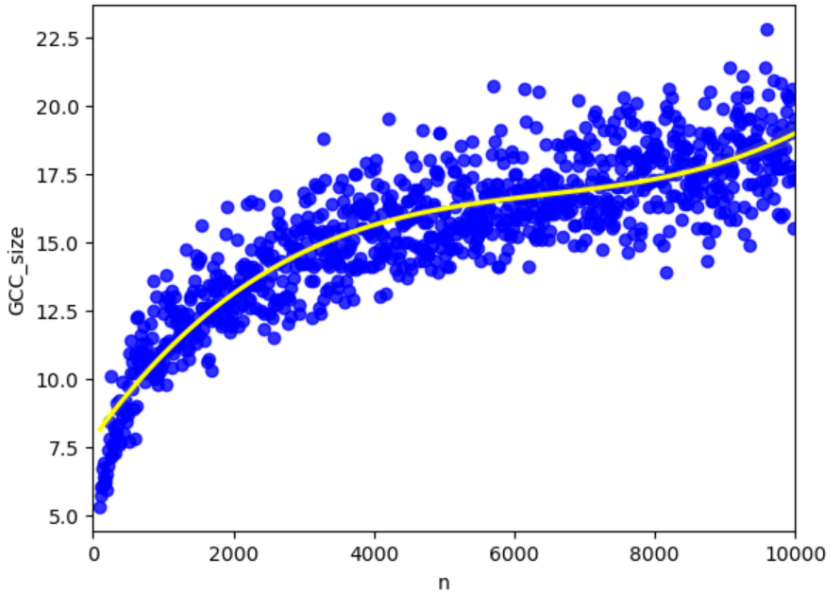


Figure 3. Expected GCC size vs n (c = 0.5)

As shown in Figure 3, a linear trend is observed when n is becoming larger. When n is smaller than 3000, the expected GCC size becomes higher, while n is bigger than 3000, the slop of the expected GCC size is getting fixed.

ii.Repeat the same for c = 1.

Answer:

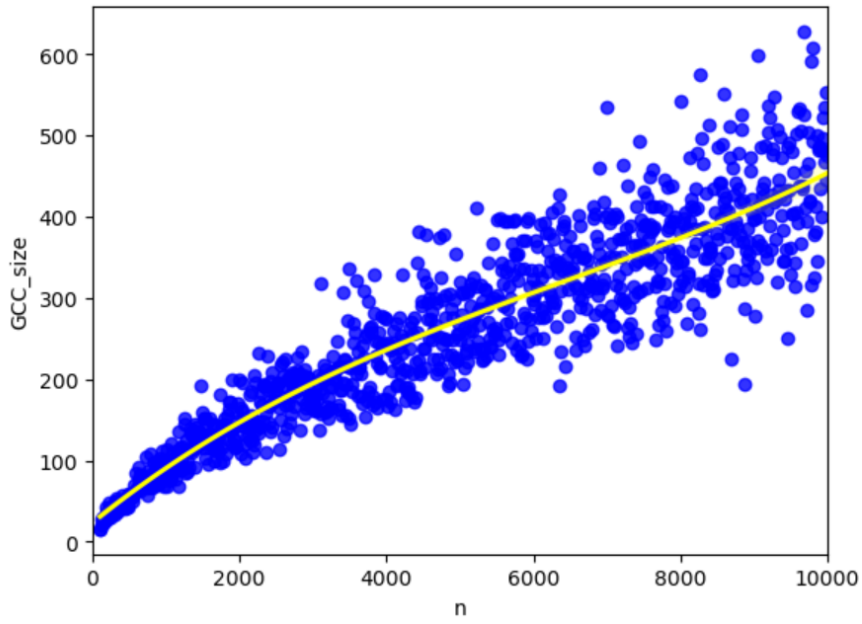


Figure 4. Expected GCC size vs n (c = 1)

As shown in Figure 4, a linear trend is observed from n = 100 to n=10000. A large slope of growing when n is small doesn't exist, compared to the curve when c = 0.5.

iii. Repeat the same for values of c = 1.15, 1.25, 1.35, and show the results for these

three values in a single plot.

Answer:

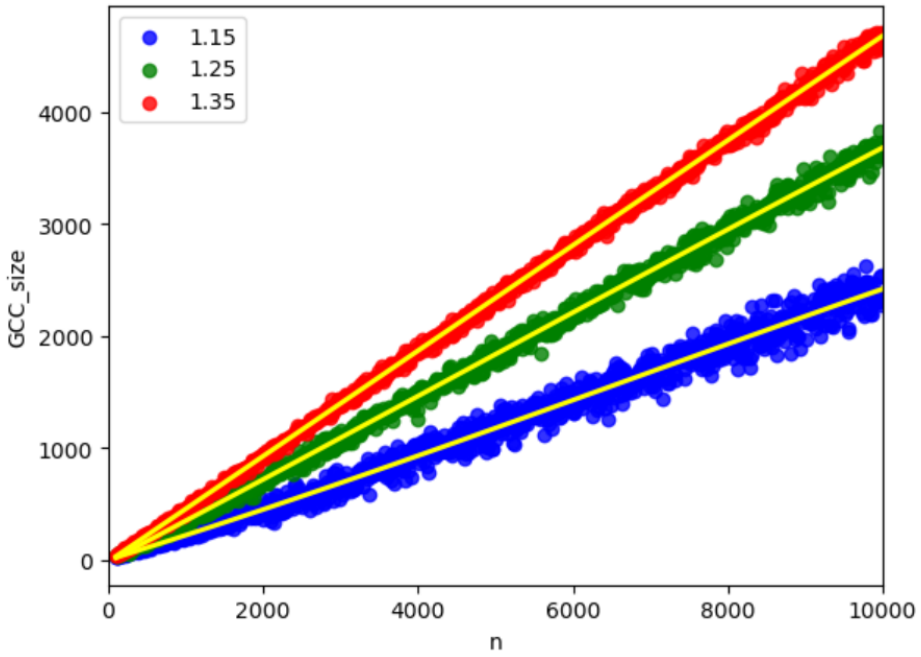


Figure 5. Expected GCC size vs n (c = 1.15, 1.25, 1.35)

iv. What is the relation between the expected GCC size and n in each case?

Answer:

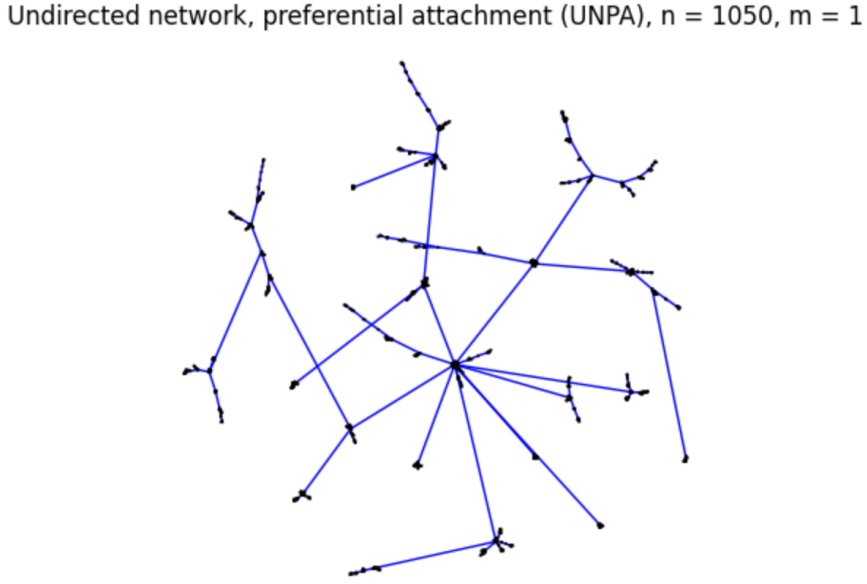
From the scatter plots, GCC size would increase as n increase, given a fixed c = np. The linear relationship is observed as c = 1.15, 1.25, 1.35. Thus, as c increases, the relationship between GCC size and n becomes more linear, and also GCC size itself is larger.

2. Create networks using preferential attachment model

(a) Create an undirected network with n = 1050 nodes, with preferential attachment model, where each new node attaches to m = 1 old nodes. Is such a network always connected?

Answer:

Yes, In a preferential attachment model, a new node is always connected to an old node, ensuring that the network is always connected. also proved in the code.




```
True
```

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

Answer:

Firstly define Assortativity, Assortativity is a measure of the tendency of nodes with similar degrees to be connected to each other. Assortativity ranges from -1 to 1, where values close to 1 indicate a high degree of assortativity and values close to -1 indicate a high degree of disassortativity.

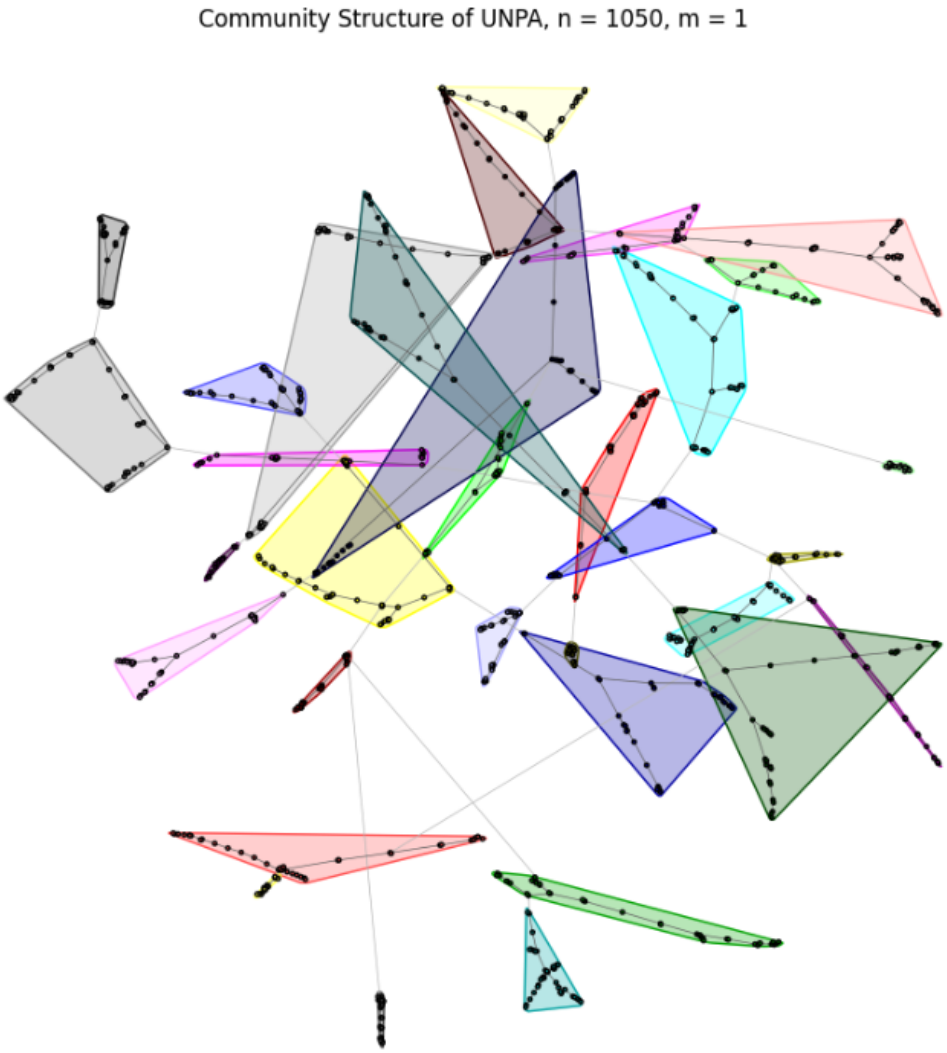
In mathematical definition, Assortativity is

$$r = \frac{\sum_i e_{ii} - \sum_i a_i b_i}{1 - \sum_i a_i b_i}$$

And for this structure,

Modularity: 0.937280591348063

Assortativity: -0.2137043156129703



(c) Try to generate a larger network with 10500 nodes using the same model. Compute modularity and assortativity. How is it compared to the smaller network's modularity?

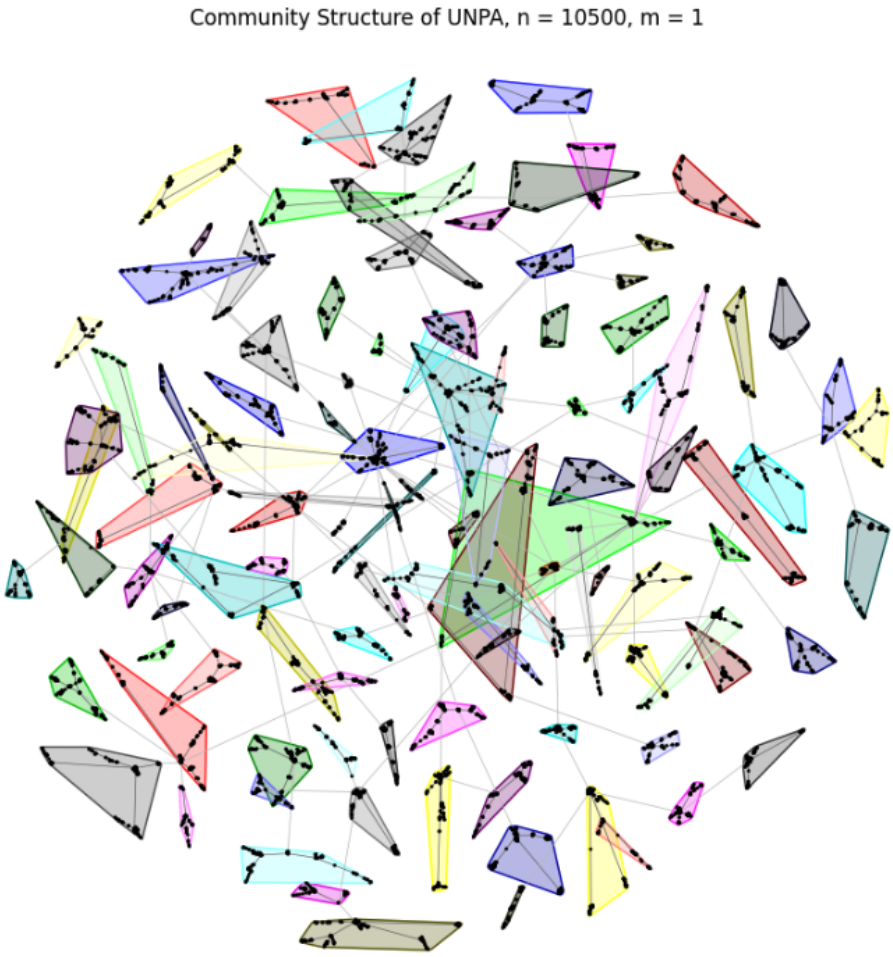
Answer:

Modularity and Assortativity are shown below.

Modularity (large network): 0.97910182003736

Assortativity (large network): -0.054613911583866595

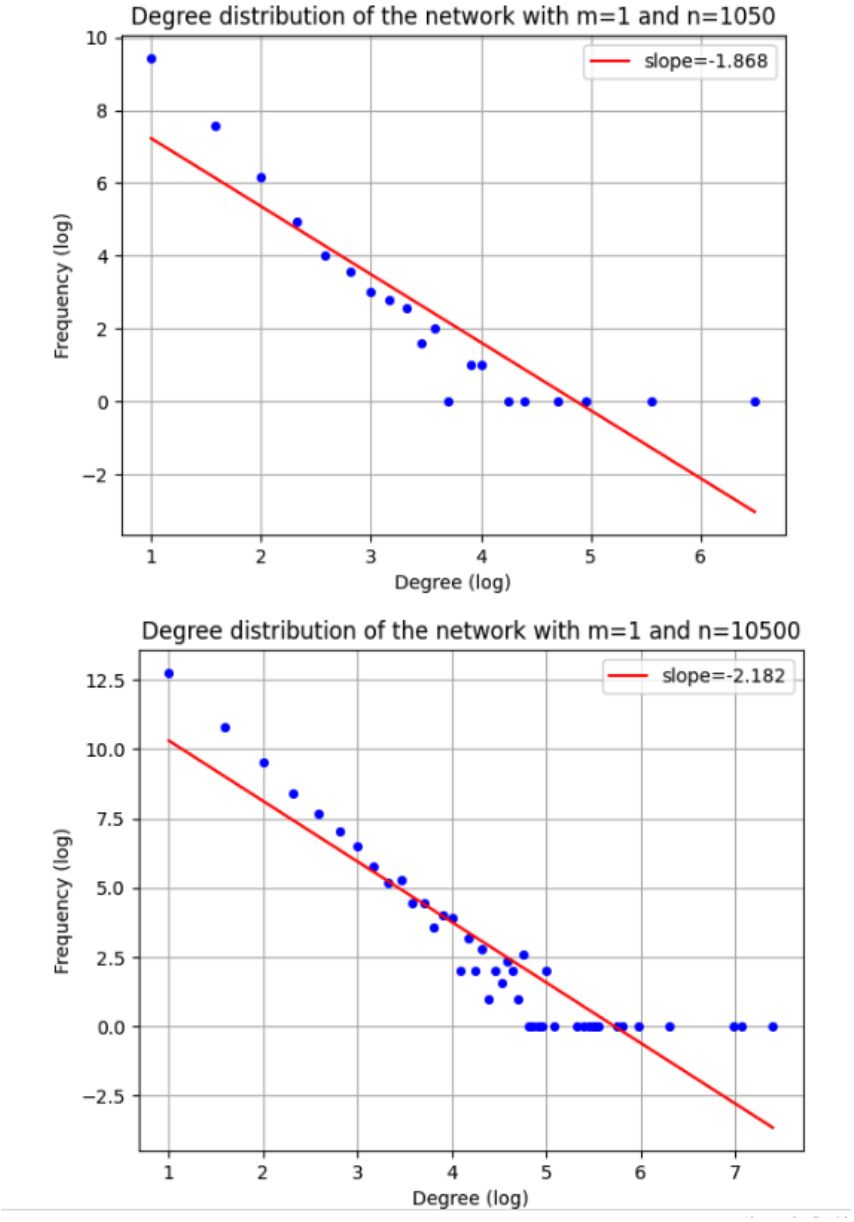
The modularity of the larger network is higher than the smaller network. This is because in In larger networks, there is typically a greater number of communities clustered together, as a result of having a higher number of nodes with higher degrees. Further, networks get larger, they tend to have more complex and diverse structures, making it more difficult to identify clear communities or modules within the network.



(d) Plot the degree distribution in a log-log scale for both n = 1050, 10500, then estimate the slope of the plot using linear regression.

Answer:

The plot is shown below. Slope with m = 1 and n = 1050: -1.868, and Slope with m = 1 and n = 10500: -2.182.

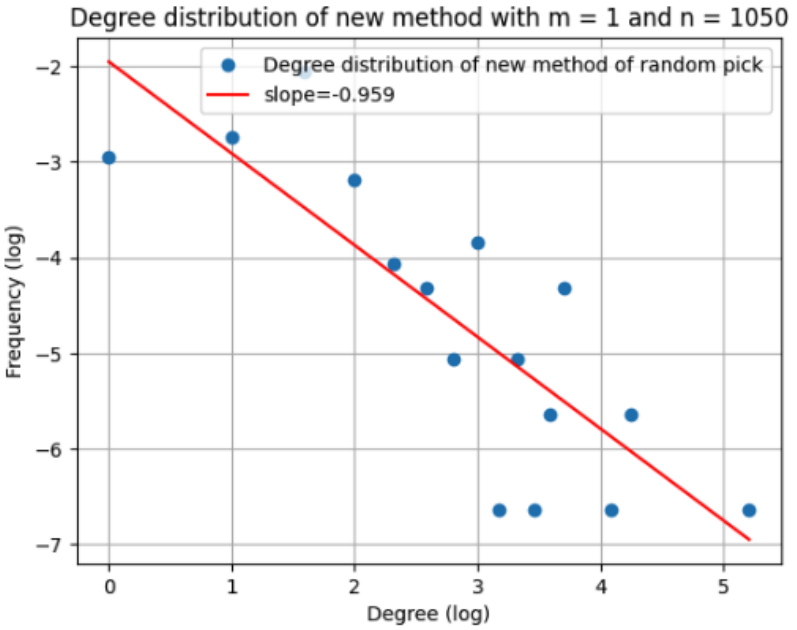


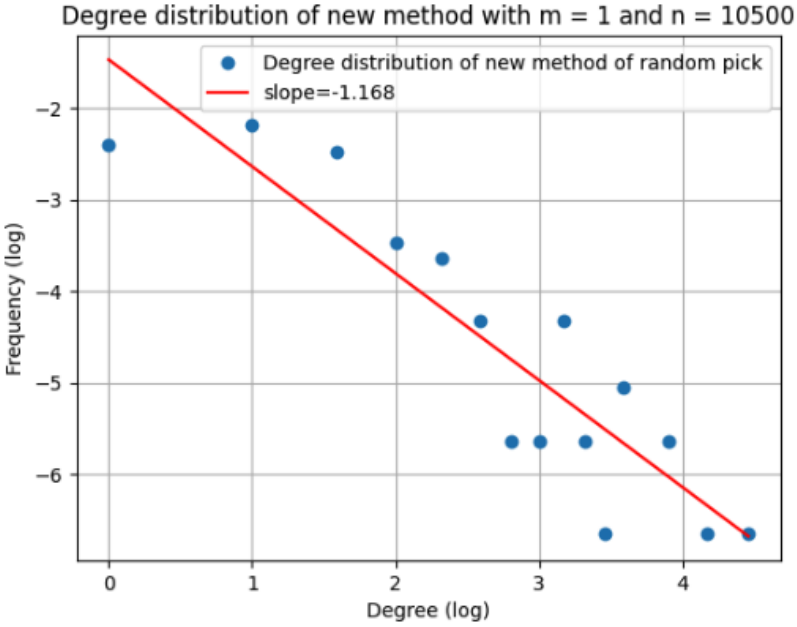
(e) In the two networks generated in 2(a) and 2(c), perform the following: Randomly pick a node i, and then randomly pick a neighbor j of that node. Plot the degree distribution of nodes j that are picked with this process, in the log-log scale.

Is the distribution linear in the log-log scale? If so, what is the slope? How does this differ from the node degree distribution?

Answer:

The plots are shown below, and the distribution is relatively linear in the log-log scale. The slope for n = 1050 and 10500, one is -0.959, the other one is -1.168. And we can see that compared to the previous node degree distribution, the slopes are significantly smaller.

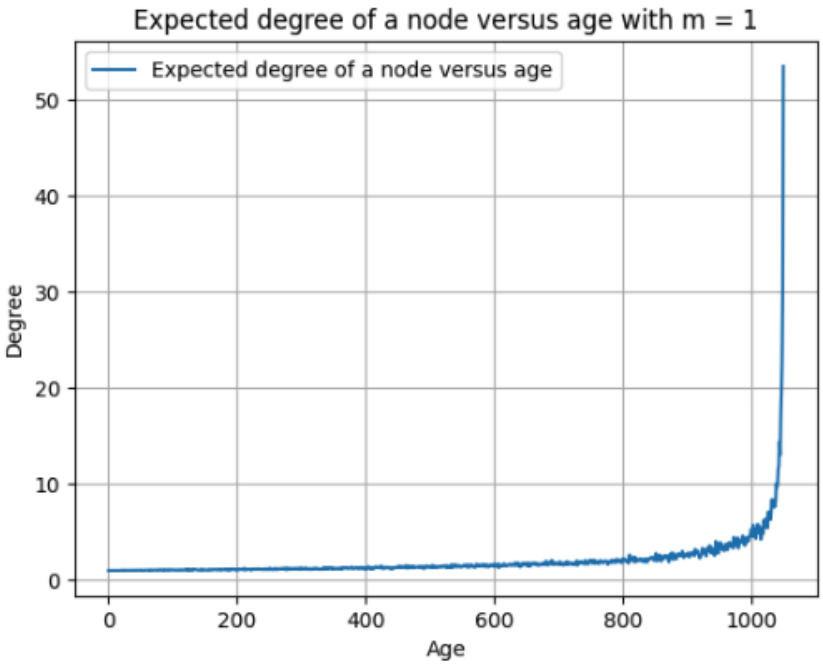




(f) Estimate the expected degree of a node that is added at time step i for $1 \leq i \leq 1050$. Show the relationship between the age of nodes and their expected degree through an appropriate plot. Note that the newest added node is the youngest.

Answer:

The relationship plot is shown below.

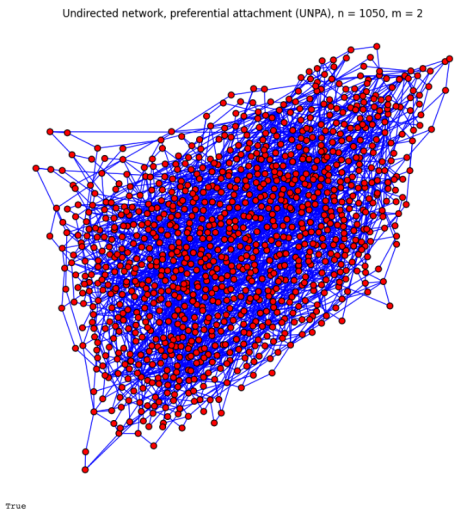


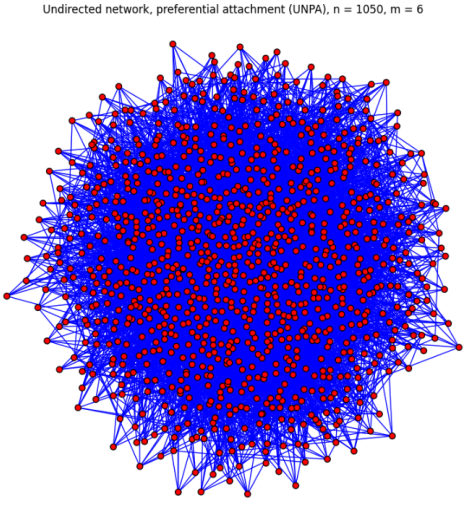
(g) Repeat the previous parts (a-f) for m = 2, and m = 6. Compare the results of each part for different values of m.

Answer:

Repeat the previous parts(a-f) for both m= 2 and m = 6

For (a), the plots are shown below. And for both m =2 and m = 6, they are all connected.



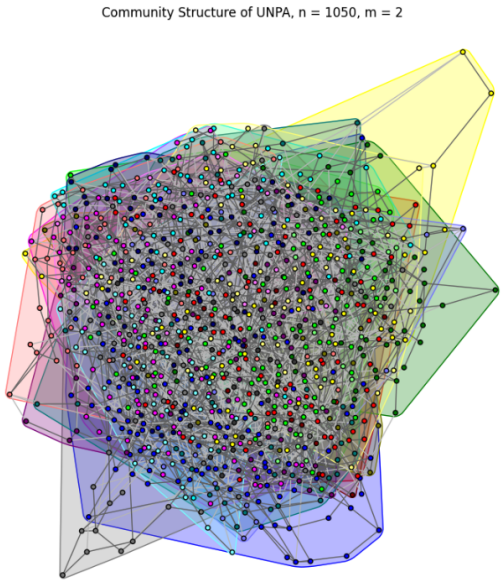


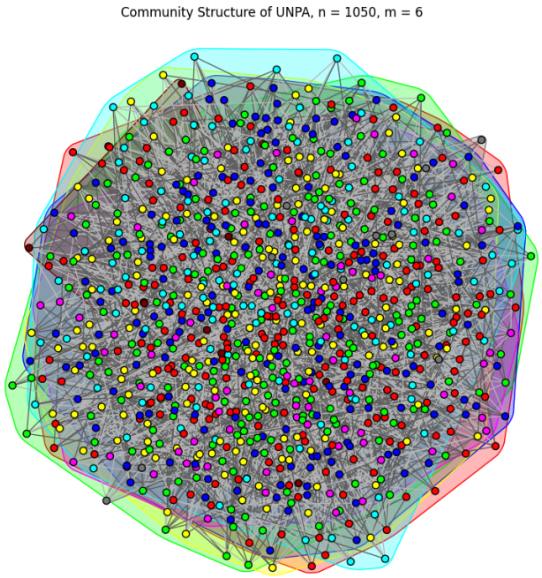
True

For (b) we observe that when m increases, the modularity decreases. and assortativity score is near to 0(or smaller) when m increases.

m =2 and n =1050, Modularity: 0.5257686969758781 Assortativity: -0.05442542142446694

m = 6 and n = 1050, Modularity: 0.24695211955022084 Assortativity: -0.01609929302812652





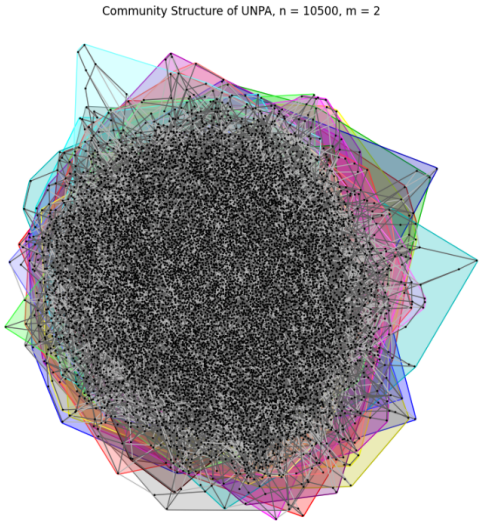
For (c) We observe that when m increases, the modularity score tends to decrease, but when n increases with the same value of m, the modularity continues to increase.and assortativity score is near to 0(or smaller) when m increases.

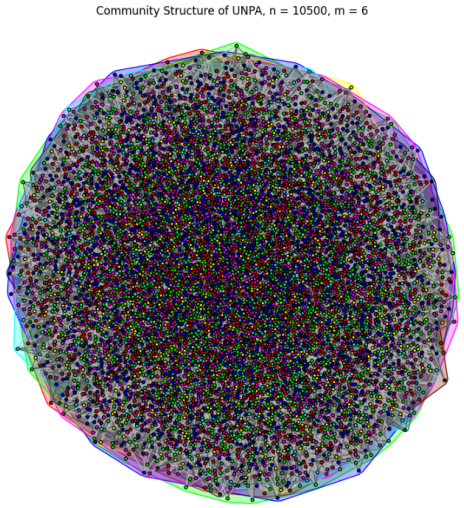
m =2 and n =10500, Modularity (large network): 0.5330937653437401

Assortativity (large network): -0.005478858614526075

m = 6 and n = 10500, Modularity (large network): 0.24769718581990943

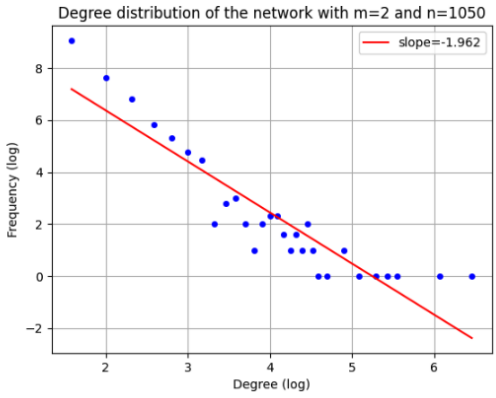
Assortativity (large network): -0.001223392101940555

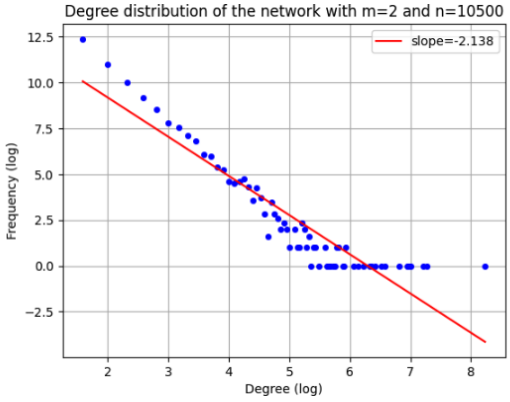




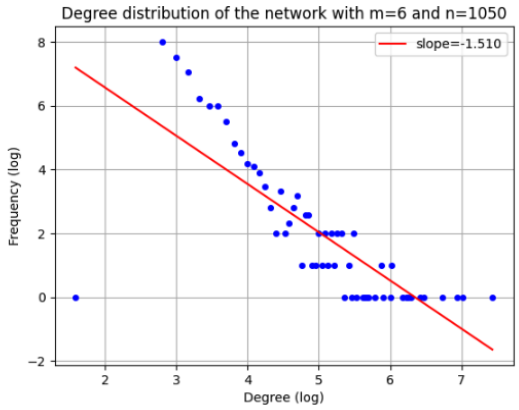
For (d) we observe that when m increases, the slope value decreases.

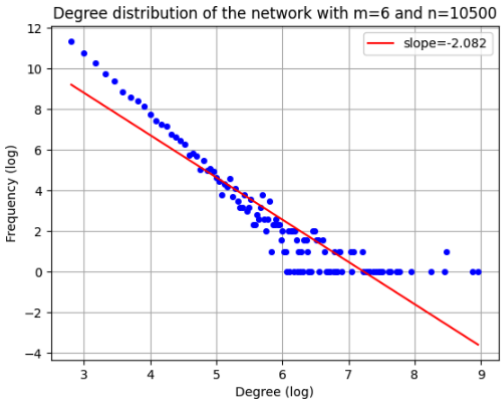
m =2 and n = 1050,10500





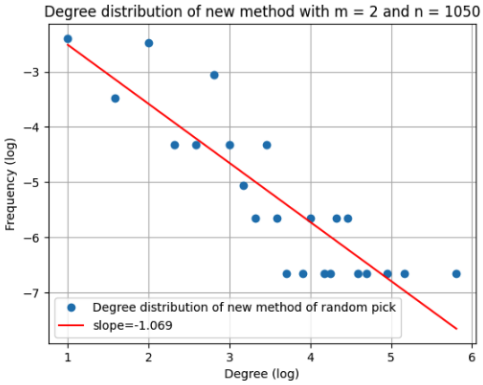
m = 6 and n = 1050,10500

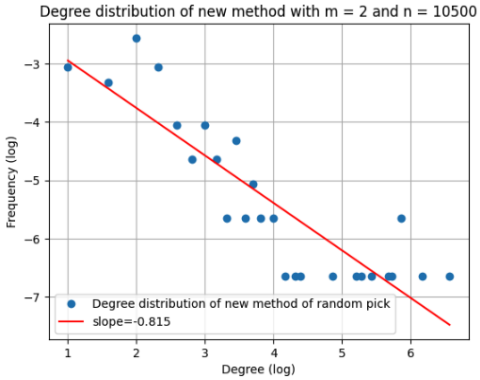




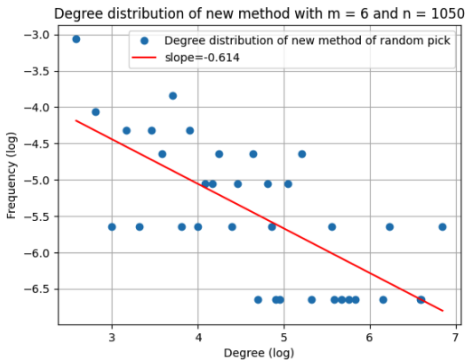
For (e), the slope absolute value decreases when m increases. And for both m=2 and m=6, their neighbor distribution slopes values are significantly less than previous degree distribution.

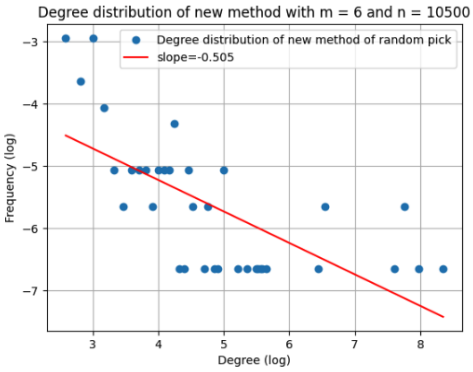
m = 2 and n = 1050,10500



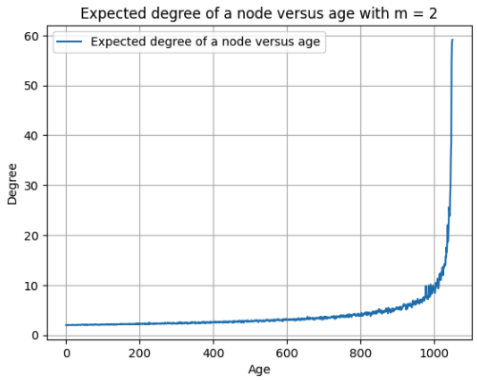


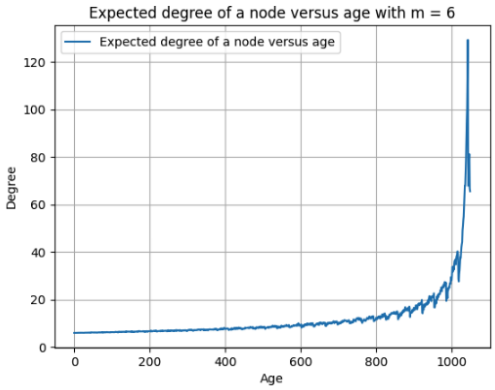
m = 6 and n = 1050,10500





For (f), We can observe that there are similar trends across different values of m. The degree increases in proportion to the increase in m.



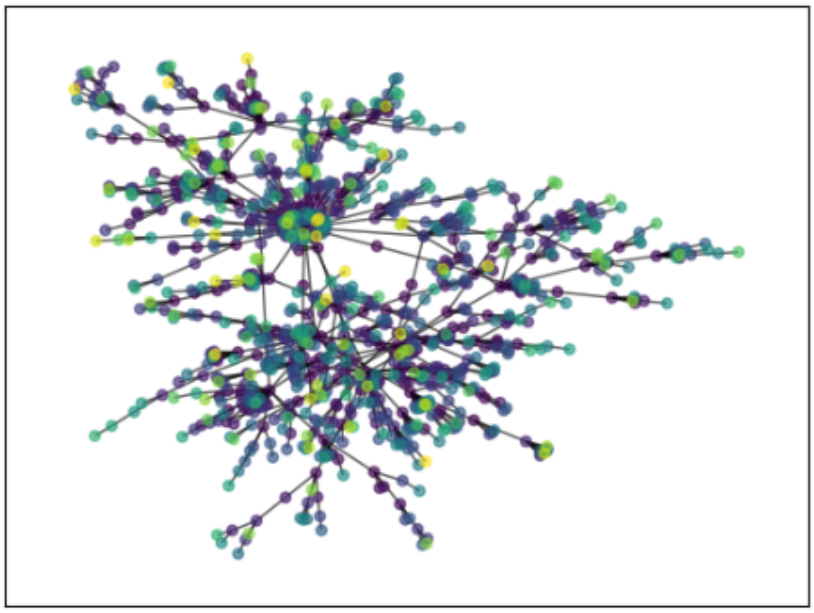


(h) Again, generate a preferential attachment network with n = 1050, m = 1. Take its degree sequence and create a new network with the same degree sequence, through stub-matching procedure. Plot both networks, mark communities on their plots, and measure their modularity. Compare the two procedures for creating random power-law networks.

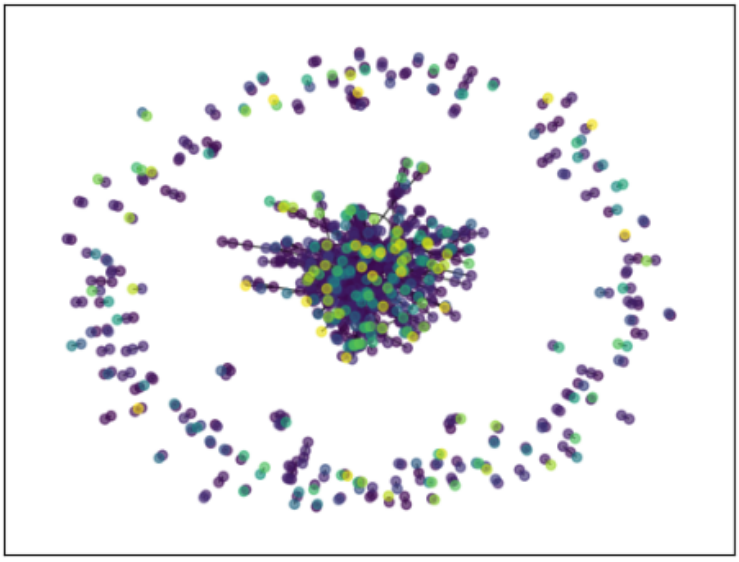
Answer: The plots are shown below. To compare the two procedures for creating random power-law networks,

Initially, we compared the results obtained from the two procedures and discovered that the graph generated from the stub-matching approach may not be connected, despite originating from a connected network. Additionally, we noted that the modularity score of the graph produced by the stub-matching method was lower. This was attributed to the network having more inter-module connections, leading to a denser plot and ultimately, a lower modularity score.

Modularity: 0.919



Modularity: 0.828



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

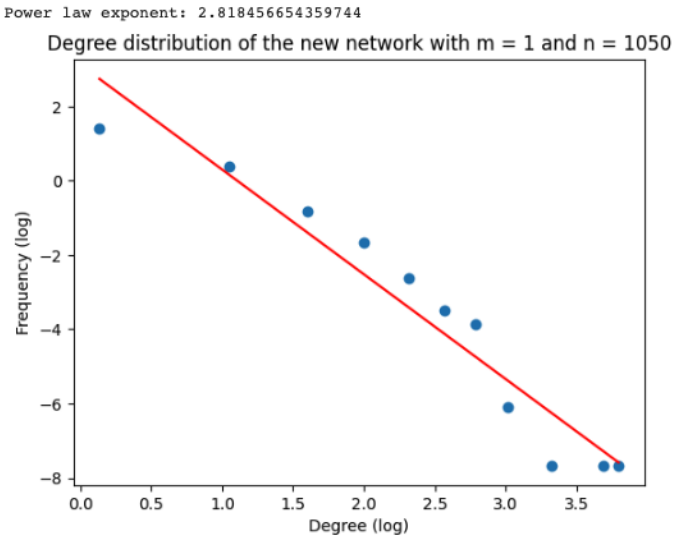
(a) Each time a new vertex is added, it creates m links to old vertices and the probability that an old vertex is cited depends on its degree (preferential attachment) and age.

In particular, the probability that a newly added vertex connects to an old vertex is proportional to:

where ki is the degree of vertex i in the current time step, and li is the age of vertex i. Produce such an undirected network with 1050 nodes and parameters m = 1, α = 1, β = −1, and a = c = d = 1, b = 0. Plot the degree distribution. What is the power law exponent?

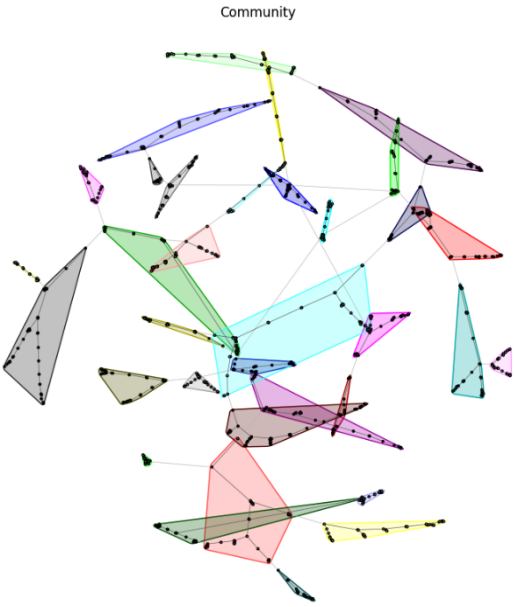
Answer:

The plot is shown below, and the power law exponent is 2.818.



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

Answer: The community structure are shown below. The modularity is `0.9370415875667141`



2. Random Walk on Networks

1. Random walk on Erdos-Renyi networks

(a) Create an undirected random network with 900 nodes, and the probability p for drawing an edge between any pair of nodes equal to 0.015.

Answer:

The generated graph is visualized as follows, with n = 900 and p = 0.015.

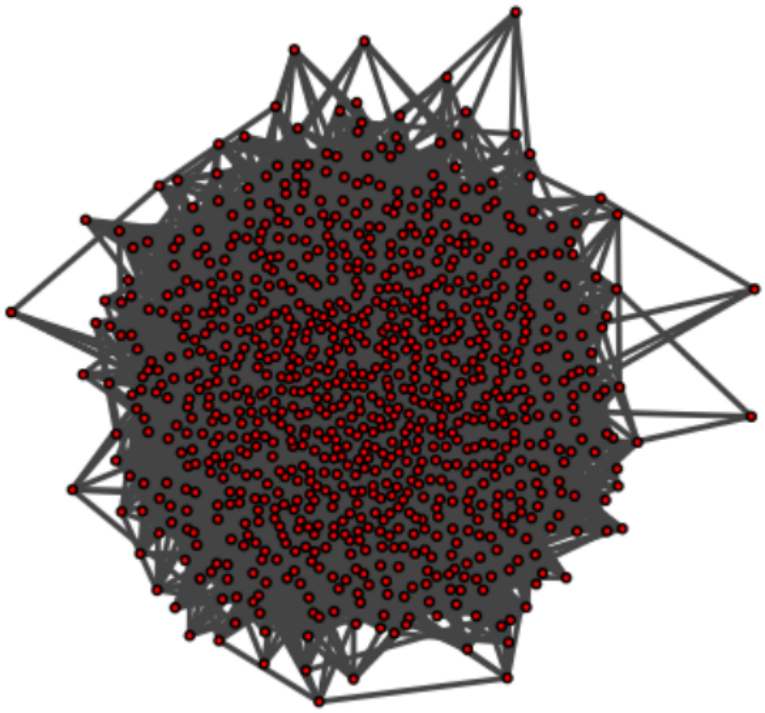


Figure 1. ER network of 900 nodes with p = 0.015

(b). Let a random walker start from a randomly selected node (no teleportation). We use t to denote the number of steps that the walker has taken. Measure the average distance (defined as the shortest path length) ⟨s(t)⟩ of the walker from his starting point at step t. Also, measure the variance σ2(t) = ⟨(s(t) − ⟨s(t)⟩)2⟩ of this distance. Plot ⟨s(t)⟩ v.s. t and σ2(t) v.s. t. Here, the average ⟨·⟩ is over random choices of the starting nodes.

Answer:

The random walk was simulated in the network's Giant Connected Component (GCC). The steps are recorded under 60, and the <s(t)> is the average shortest distance of 100 walks.

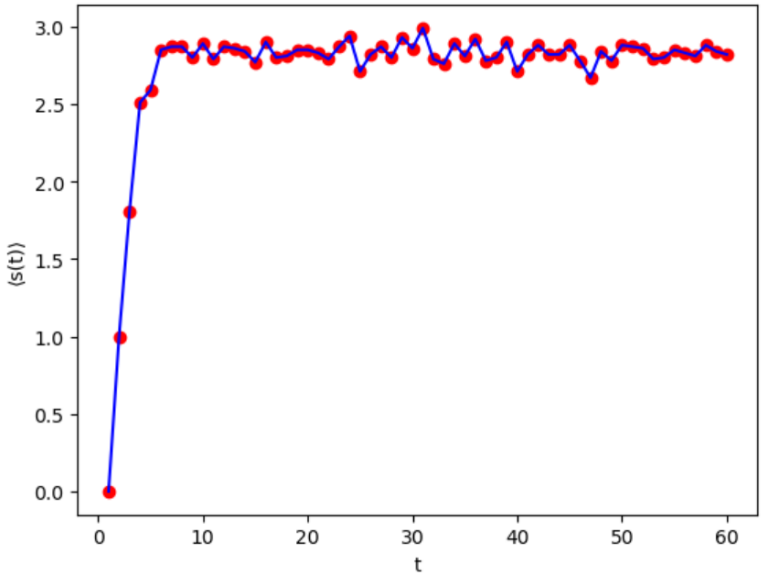


Figure 2. Line chart of the average distance

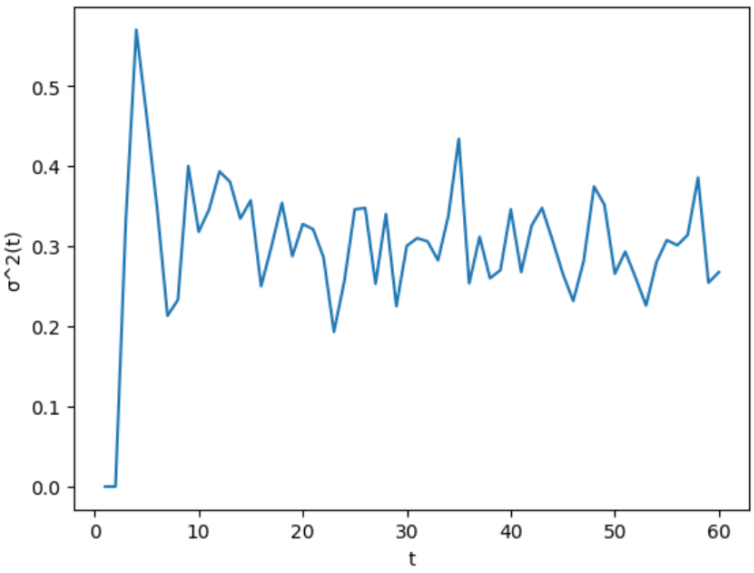


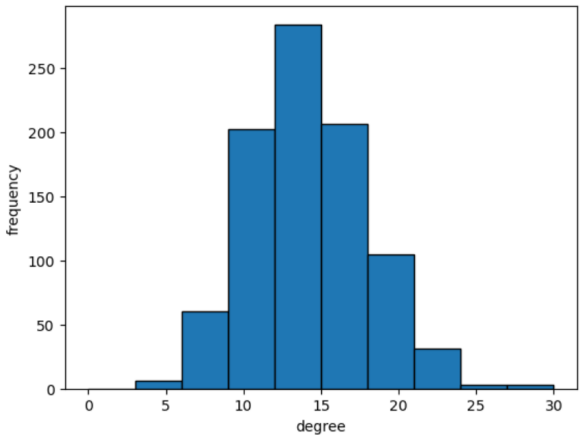
Figure 3. Variance plot of shortest path length

It is observed that the variance of the average value of the shortest distance increases until it reaches a steady value when t = 10. The average is close to 2.7. The variance first increases as t is smaller than 5 and reaches the highest. Then it gradually drops to 0.3 and keeps the value after the step is more than 10.

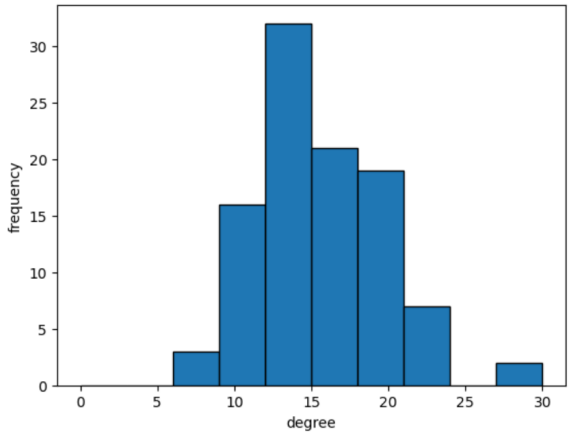
(c) Measure the degree distribution of the nodes reached at the end of the random walk.

How does it compare to the degree distribution of graph?

Answer:



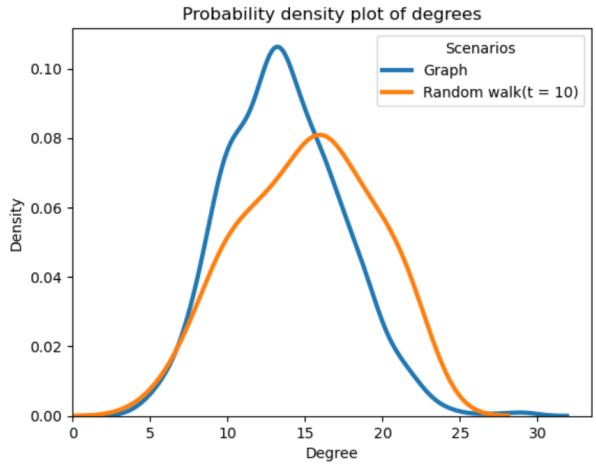
(a) Degree distribution of the original network



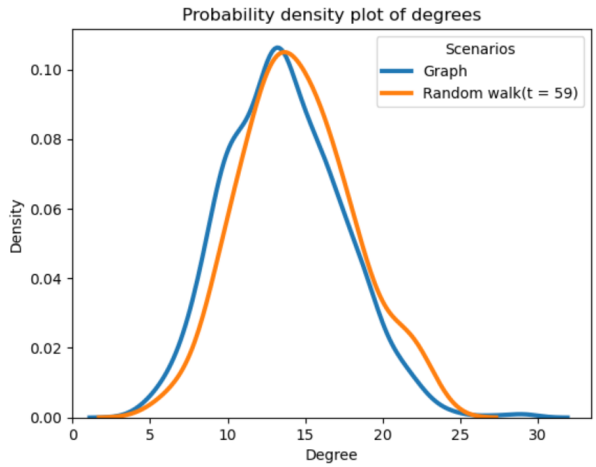
(b) Degree distribution of the nodes at the end of the random walk

Figure 4. Degree distribution

It is observed that both distributions follow the binomial distribution, and the patterns look the same.



(a) Degree distribution comparison (t = 10)



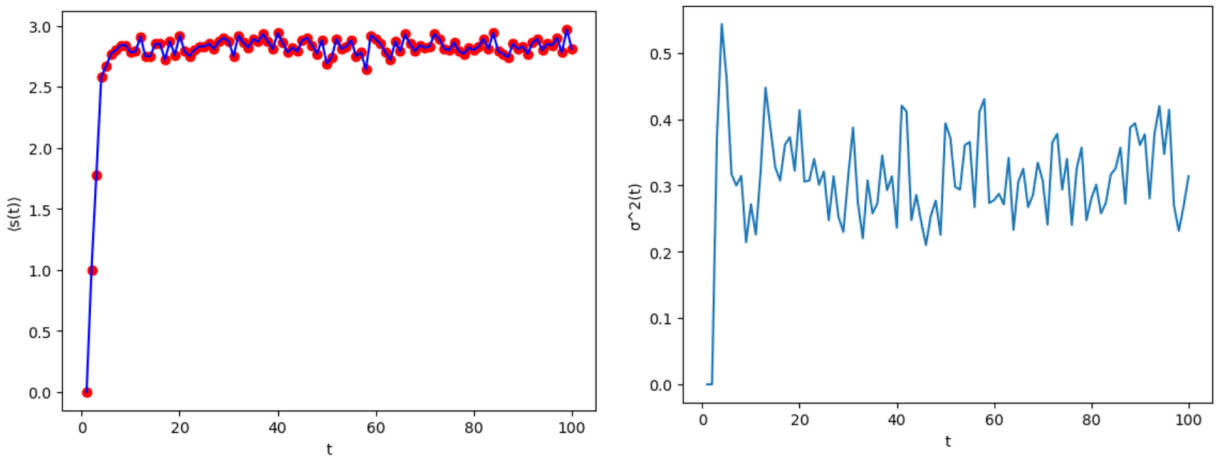
(b) Degree distribution comparison (t = 60)

Figure 5. Probability density distribution of degree in the raw graph and from random walk

As shown in this graph (Figure 5(a).), after 10 steps, the distribution of the nodes' degree is different from that of the graph. But after 60 steps as in Figure 5(b), the distribution of the degree is much similar to the degree distribution of the raw graph. Both the degree distribution of the graph and the nodes after the random walk follow the binomial distribution.

(d) Repeat 1(b) for undirected random networks with 9000 nodes. Compare the results and explain qualitatively. Does the diameter of the network play a role?

Answer:



(a) ⟨s(t)⟩ vs t (b) σ^2(t) vs t

Figure 6. Distance of random walk (n =9000, t = 100)

Table 1. Comparison of the random walk results

	Diameter	t (steady)	⟨s(t)⟩	σ^2(t)
N = 900	5	10	2.70	0.33
N = 9000	3	6	2.80	0.32

Interestingly, when the network size (n) increases, both the number of steps required to reach the steady state, the values of ⟨s(t)⟩ and σ^2(t) decrease. This implies that a larger network reaches the steady state more quickly, and at convergence, there is a smaller variation in the shortest distance to the starting node. Although this may seem counterintuitive, it is essential to note that as the network size increased from 1000 to 10000, the network diameter reduced from 5 to 3. This indicates that, on average, the network is denser around the starting node, leading to fewer steps needed for convergence.

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

(a) Generate an undirected preferential attachment network with 900 nodes, where each new node attaches to m = 1 old nodes.

Answer:

Use the following code to generate the network where N is the number of nodes

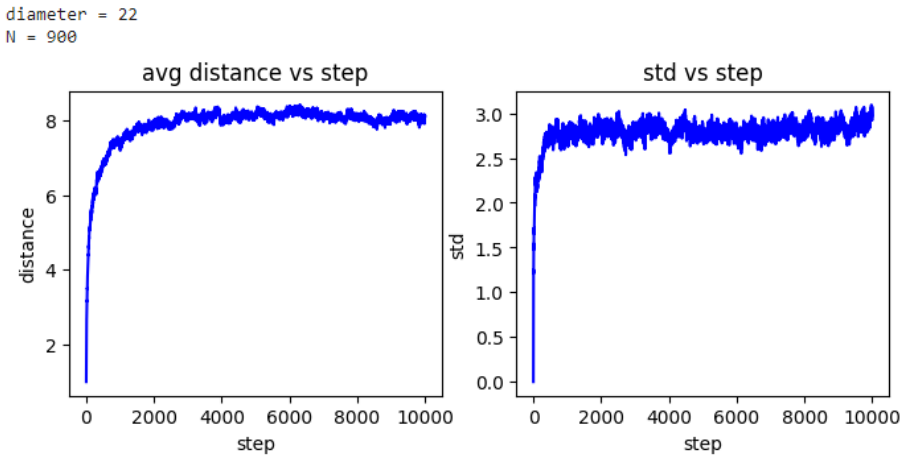
```
g2_2 = igraph.Graph.Barabasi(N, 1, directed=False)
```

(b) Let a random walker start from a randomly selected node. Measure and plot ⟨s(t)⟩

v.s. t and σ2(t) v.s. t.

Answer:

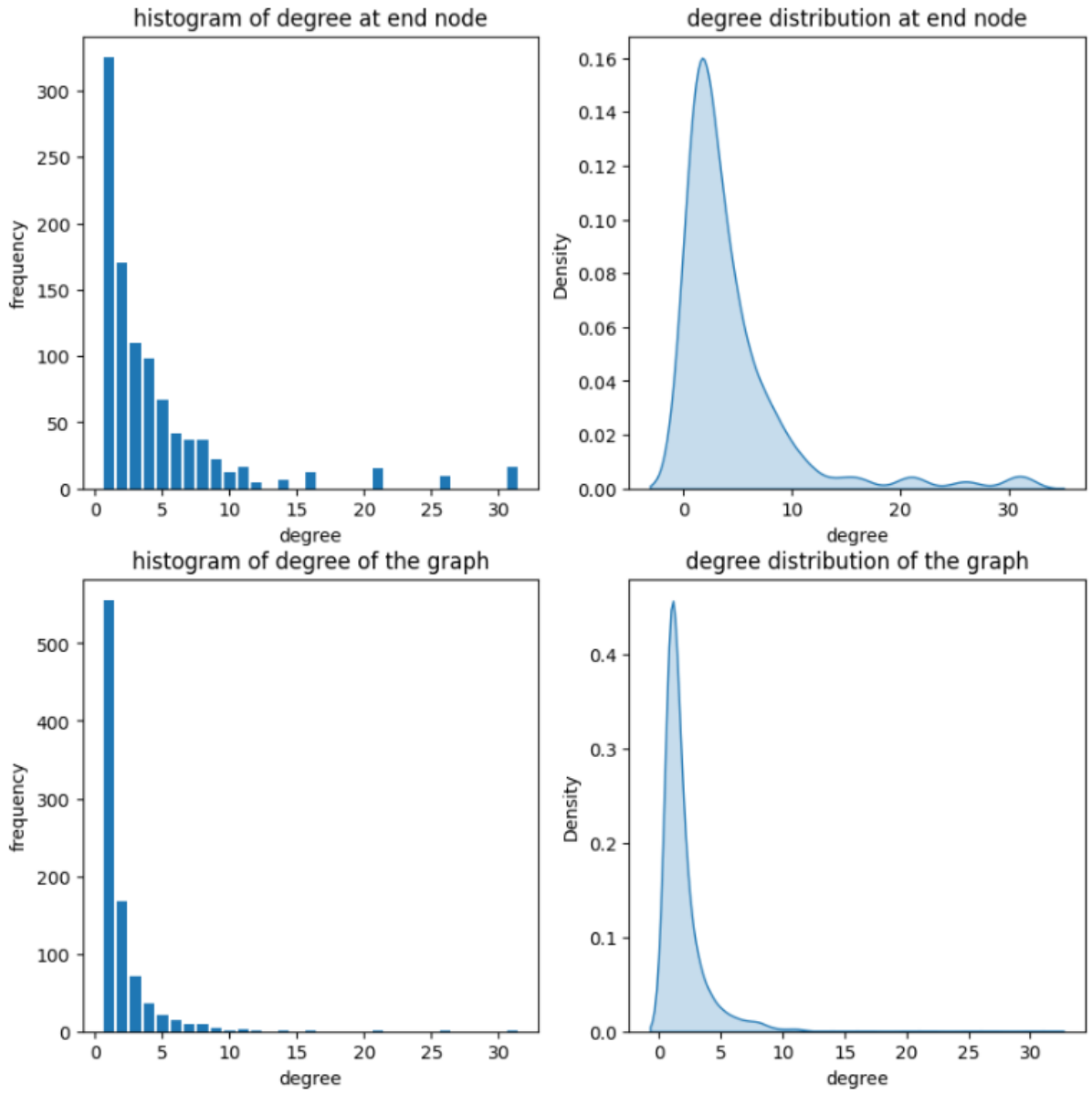
In this question we let a random walker start at 500 different locations and each time walk 1000 steps. For each step, measure the distance and calculate its average and standard deviation and get the following plot:



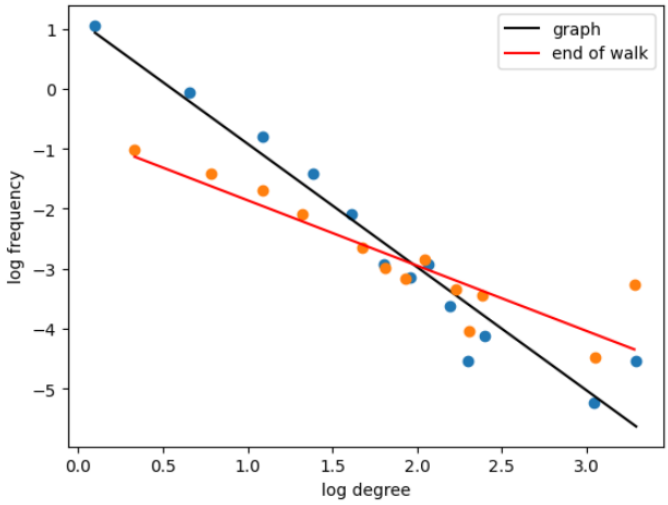
(c) Measure the degree distribution of the nodes reached at the end of the random walk on this network. How does it compare with the degree distribution of the graph?

Answer:

The following figure shows the degree distribution of the nodes reached at the end of the random walk, compared with the degree distribution of the graph. We can see that most of the vertices has very low degree, but at the end of the walk, the probability of reaching a high-degree vertex is higher than its probability appears in a graph. This is because the walker is more likely to walk to the central node of the network, so the frequency of the high degree nodes at the end of each walk is higher.



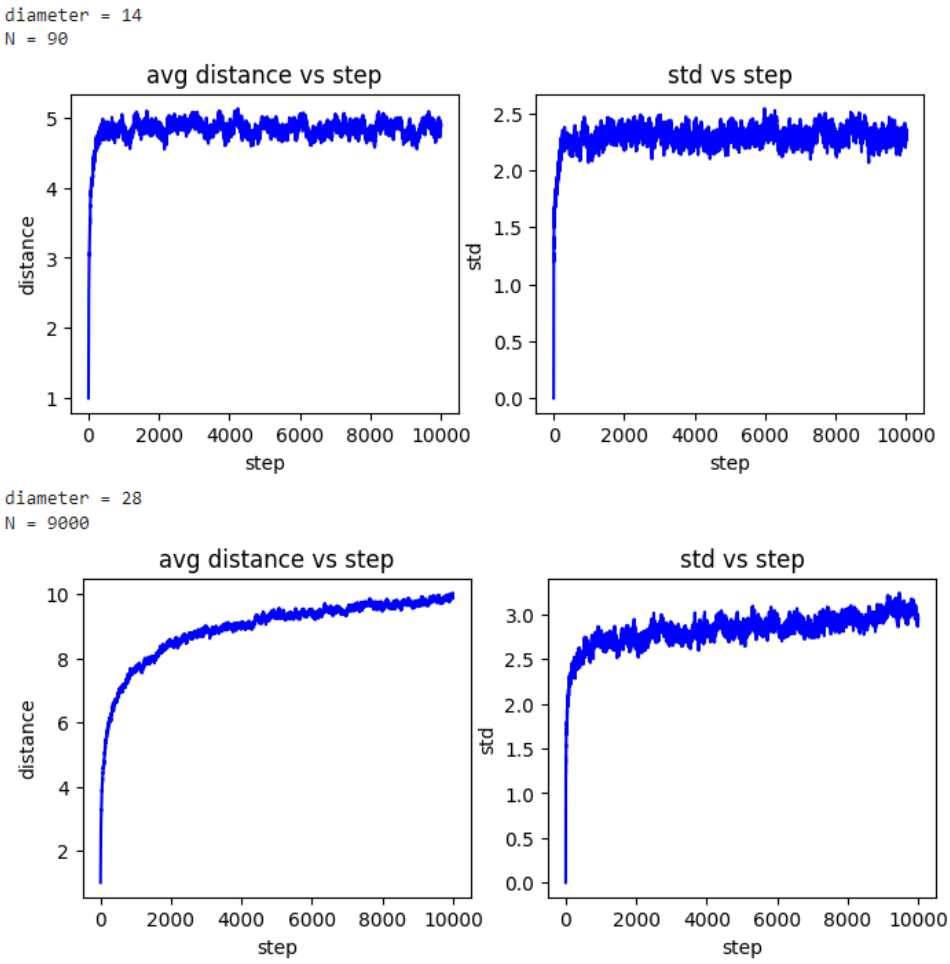
We can also plot the log-log figure of the degree distribution. The following figure shows the log-log degree distribution of the nodes reached at the end of the random walk, compared with the degree distribution of the graph. We can see that the slop of the end of the walk is more gently. The reason is because the end-of-walk node has a high frequency at high degree, same as above.



(d) Repeat 2(b) for preferential attachment networks with 90 and 9000 nodes, and m = 1. Compare the results and explain qualitatively. Does the diameter of the network play a role?

Answer:

The following figure shows the distance and standard deviation against step for N is 90, 9000. Compared with the result of N = 900 nodes, We can see that for a larger graph, the walker can get farther from the start point, and the std is also larger. We can use the g.diameter() function to get the diameter of the networks, the diameter for network with N = 90, 900, 9000 is 14, 22, 28, respectively. So we can know that unlike the Erdos-Renyi networks, the preferential attachment network grows larger as the nodes increases, so the distance, std, and diameter all increases.



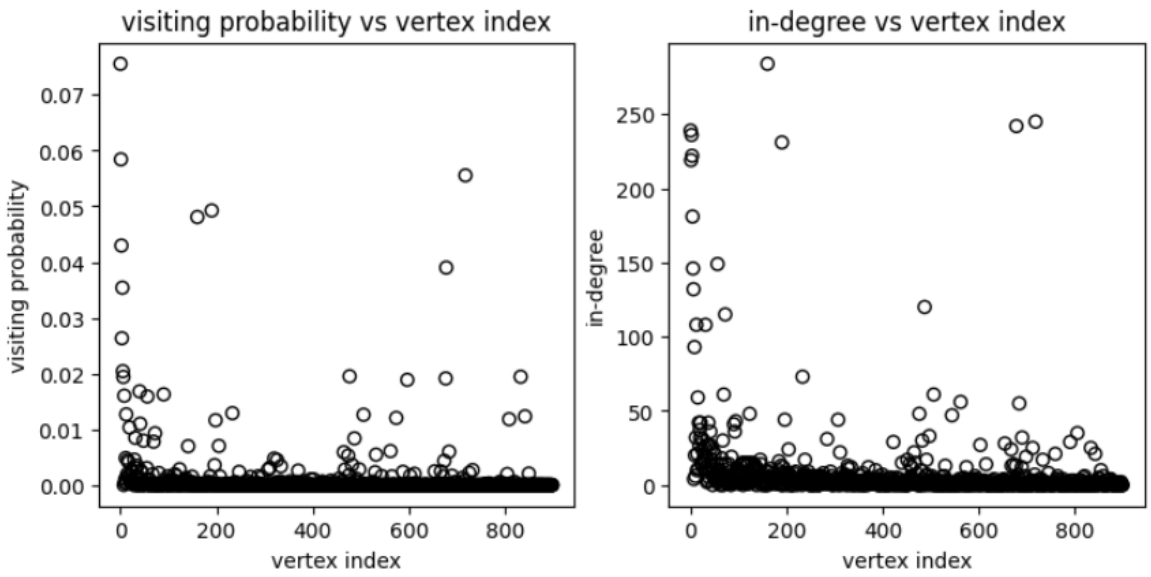
3. PageRank

(a) Measure the probability that the walker visits each node. Is this probability related to the degree of the nodes?

Answer:

In this question we make 1000 random walker and each of them walk 1000 times to get a total of 1e6 visited nodes and measure the probability of visiting each of the vertices.

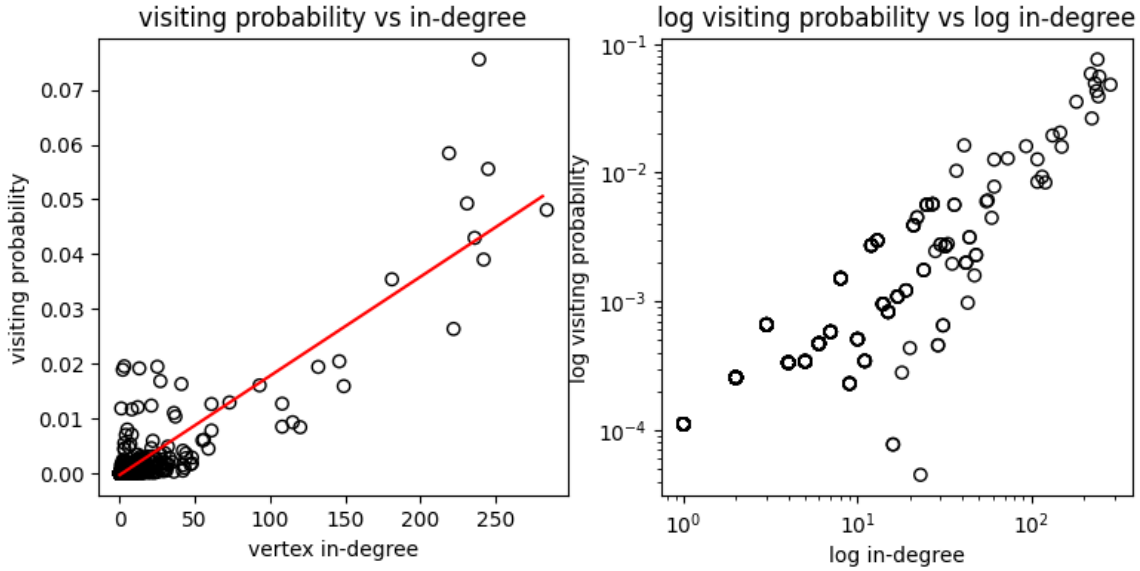
The follow figure shows the probability of visiting each vertex and the in-degree of each vertex. We can see that there are several outstanding vertices have a verge large(>200) in-degree, and correspondingly, those vertices have a high(>0.03) visiting probability. Those vertices have index of [0,1,2,3, 160,190,678,718]. Clearly, the vertices 0~3 are the original “Super nodes” of the preferential attachment network, and the other four nodes are new “Super nodes” generated by shuffling the indexes to avoid black hole.



Now we perform the correlation analysis: We calculate the Pearson R Result using the function:

```
stats.pearsonr(p_visited, metric)
```

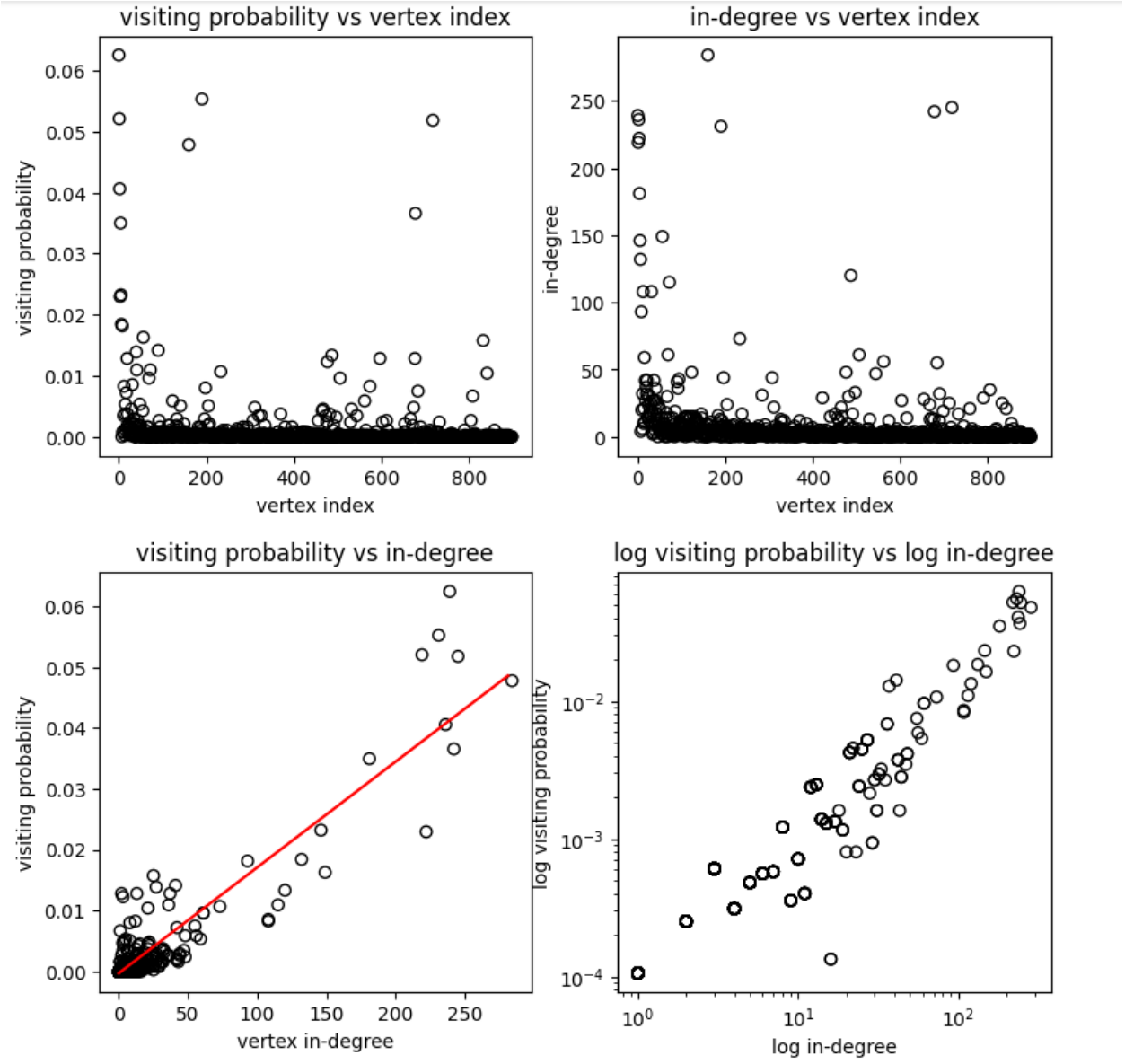
where the p_visited is the visiting probability and the metric is in-degree, and find that the PearsonRResult is 0.89 and pvalue is 0.0. which indicates a strong linear relationship. The following figure shows the plot of visiting probability vs nodes in-degree, from the plot e can also see a strong linear relationship.



(b) In all previous questions, we didn't have any teleportation. Now, we use a teleportation probability of α = 0.2 (teleport out of a node with prob=0.2 instead of going to its neighbor). By performing random walks on the network created in 3(a), measure the probability that the walker visits each node. How is this probability related to the degree of the node and α ?

Answer:

Same with 3(a), we make 1000 random walker and each of them walk ideally 1000 times. But at each step, the walker has a probability of 0.2 to stop. So at the end we only get 3744 visited nodes, which means the expectation of the total step of the random walk is 3.7. The step can be affected by the teleportation probability α. with a higher α, the average length will be much smaller. However, the probability of visiting each nodes doesn't affected by α, since the probability of visiting each nodes still mainly correlated with the in-degree of the node. At tis time we still draws the same 4 figures as 3(a), and the PearsonRResult is 0.92.



4. Personalized PageRank

(a) Suppose you have your own notion of importance. Your interest in a node is proportional to the node's PageRank, because you totally rely upon Google to decide which website to visit (assume that these nodes represent websites). Compare the results with 3(a).

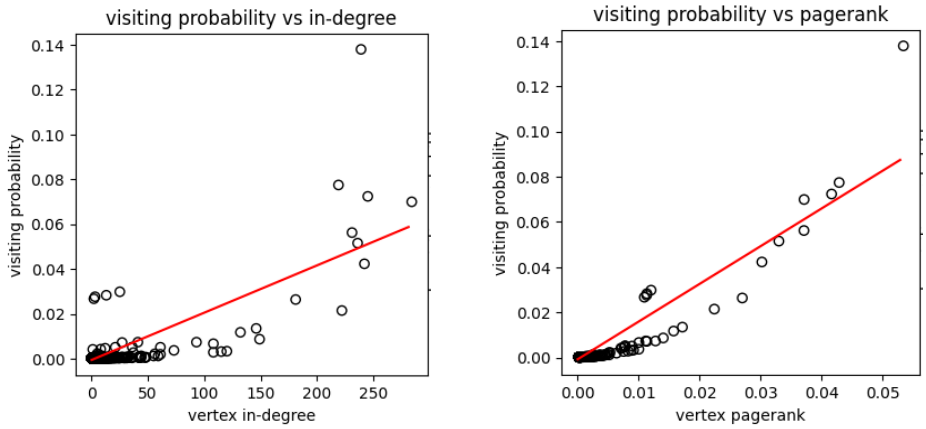
Answer:

In this question, at each step, the choice of next node is proportional to the neighbor's pagerank score, and the teleportation is to choose a random node in the whole graph. So after 1000 walks of 1000 steps, we still get 1e6 visited nodes. We first analyze the visiting probability vs vertex in-degree as we did in 3(a) and find the PearsonRResult is only 0.78, which indicates the correlation is less strong, that is because in each step, the transition probability is proportional to neighbor's pagerank score, other than random choosing a neighbor. So the overall visiting probability is less linearly correlated to vertex in-degree. So we analyze the visiting probability vs vertex pagerank score. We calculate the pagerank score with the following code:

```
pagerank_score = ig.Graph.pagerank(g, damping=0.8)
```

where damping 0.8 means there is a probability or 0.2 to reset the random walk to a uniform distribution, which is the same as our strategy. The new PearsonRResult is 0.93, indicates a strong linear relationship.

The following figure are visiting probability vs in-degree and visiting probability vs pagerank, we can also see that the visiting probability has a better linear correlation with pagerank.

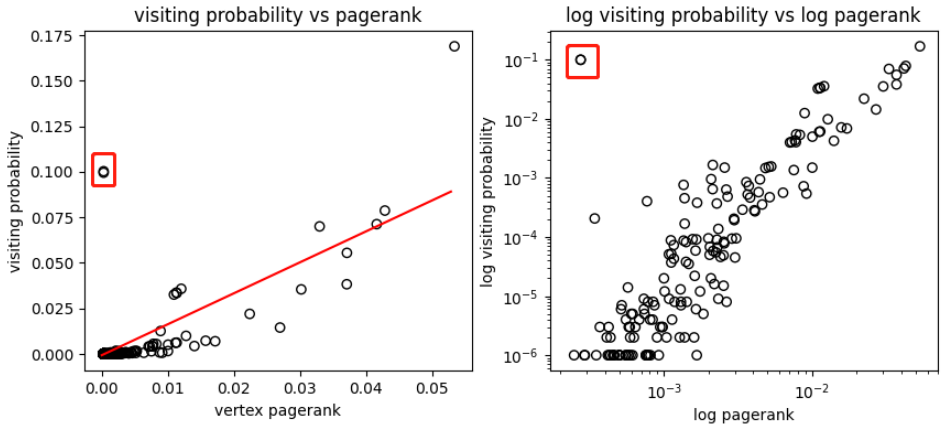


(b) Find two nodes in the network with median PageRanks. Repeat part 4(a) if teleportations land only on those two nodes (with probabilities 1/2, 1/2). How are the PageRank values affected?

Answer:

In this question, the teleportation is set to two selected nodes and the nodes are selected by take the median of the pagerank score. These two nodes are selected as index 63 and 263. The

following figure shows the visiting probability vs original pagerank score where we assume that people's interest in all nodes are the same. So we can observe two abnormal points (indicated by reb box below). These two points have a very low pagerank score but the probability of visiting those vertex is 0.1 (0.2 teleport probability on two nodes). As a result, the PearsonRResult between the visiting probability and pagerank is decreased to 0.75.



(c) More or less, 4(b) is what happens in the real world, Can you take into account the effect of this self-reinforcement and adjust the PageRank equation?

Answer:

To address our new assumption, we need to alter the way of computing pagerank score, we use the following code to calculate pagerank which considers the two trusted nodes:

```
ig.Graph.personalized_pagerank(g, damping=0.8, reset_vertices=[63, 263])
```

and repeat 4(b), we can find that the outstanding points are gone because those nodes have a new pagerank score as 0.1, and the new PearsonRResult between the visiting probability and pagerank is 0.91, indicating a stronger linear relationship.