RESEARCH REPORT I

July 11, 2019

Shiqi (Freda) Huang

Contents

1 Introduction

The project is about designing algorithms that calculate different variables (specifically diameter, cluster coefficient and degree distribution) which describe the properties of an undirected graph. It is worth noted that calculating cluster coefficient involves two very important steps: getting degeneracy ordering of the graph and counting triangles based on the ordering. And the network model I will be testing my algorithm on is Erdos-Renyi Model.

2 Diameter

2.1 Pseudocode

```
get_diameter(graph)
    last_dmax<-0;
    current_dmax<--1;
    init_vertex <- random vertex from graph
    furthest_vertex,current_dmax <- BFS(ini_vertex)
    //BFS return furthest vertex from initial node and eccentricity of th
    while current_dmax > last_dmax do
       ini_vertex<-furthest_vertex //start from the furthest vertex
       last_dmax<-current_dmax
       furthest_vertex,current_dmax=BFS(ini_vertex)
    result<-last_dmax
```

2.2 Description

The diameter function used heuristic idea II provided by professor. Roughly speaking, it repeatly run BFS on the furthest node discovered by previous BFS run.

2.3 Observation

During the testing runs, I found out that it only takes few BFS runs to get the diameter. So performance wise, it is pretty decent.

3 Cluster Coefficient - Degeneracy Ordering

3.1 Pseudocode

```
{Taken from Lecture slide , and modified}
Initialize an output list ,L, to be empty.
Compute a number,dv, for each vertex v in G, which is the number of
neighbors of v that are not already in L.
Initialize an array D such that D[i] contains a list of the vertices
v that are not already in L for which dv=i
Let Nv be a list of the neighbors of v that come before v in L.
Initially , Nv is empty for every vertex v.
//Nv contains pairs of neighbors whose degree are greater than v
Initialize k to 0
Repeat n times:
    Let i be the smallest index such that D[i] is nonempty.
    //select smallest degree vertex in the list
    Set k to max(k, i).
    D[i]. Add v to the beginning of L and remove it from D[i].
    Mark v as being in L. Select a vertex v from L .
    For each neighbor w of v not already in L
    Subtract one from dw
    Move w to the cell of D corresponding to the new value of dw
    Add w to Nv
result <- (L,N)
```

3.2 Description

Degeneracy Ordering is essentially a greedy algorithm. it repeatedly find and remove the vertex with smallest degree, pushing it to the end of the list. The resulting list gives us the degeneracy ordering.

4 Cluster Coefficient - Triangle Counting

4.1 Pseudocode

```
{Taken from lecture slide, modified}
L,N <- Degeneracy_Ordering(graph)
Process the vertices according to this ordering, L:
For each vertex, v:
    Nv <- N[v]  //Nv contains all nodes that is earlier in the ordering t
    for each pair u,w in Nv:
        If (u,w) is an edge in the graph:
            add one to the triangle count.
```

4.2 Description

The heustic of the algorithm is to take all possible pairs from neighbors for each vertex and check if they are connected which forms a triangle with the vertex.

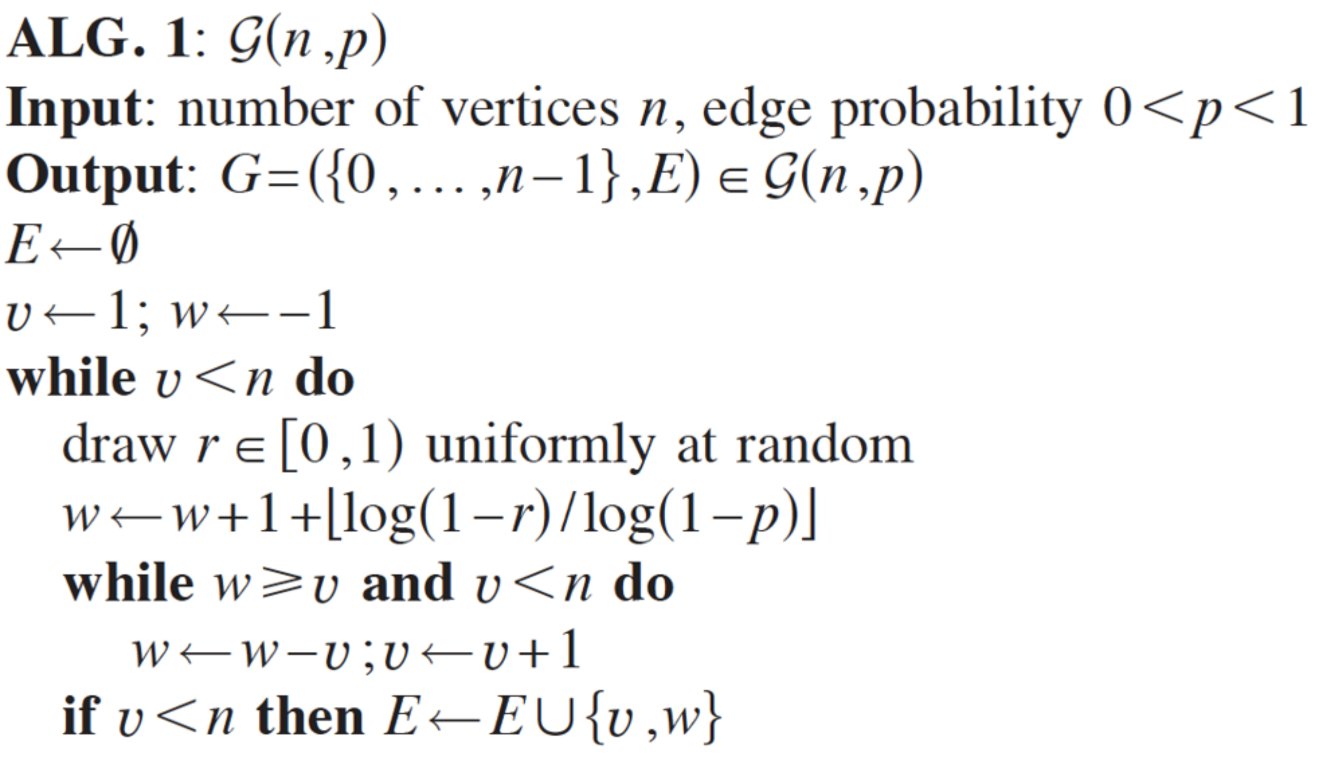
4.3 Observation

There are many ways to implement this, but the Triangle counting algorithm used here utilizes degeneracy ordering, which ensures that during the process, each triangle would be only counted once. This approach saves a lot of time by avoiding the calculation of duplicate triangles.

5 Erdos-Renyi Model

5.1 Pseudocode

<taken from lecture slide>



```
**ALG. 1**: $\mathcal{G}(n,p)$
**Input**: number of vertices $n$, edge probability $0<p<1$
**Output**: $G=(\{0,\ldots,n-1\},E)\in\mathcal{G}(n,p)$
$E\leftarrow\emptyset$
$v\leftarrow 1$; $w\leftarrow -1$
**while** $v<n$ **do**
    draw $r\in[0,1)$ uniformly at random
    $w\leftarrow w+1+\lfloor\log(1-r)/\log(1-p)\rfloor$
    **while** $w\geqslant v$ **and** $v<n$ **do**
        $w\leftarrow w-v$; $v\leftarrow v+1$
    **if** $v<n$ **then** $E\leftarrow E\cup\{v,w\}$
```

5.2 Description

This is a faster approach to generate a random Erdos-Renyi Graph. It avoids the need to generate 0 bit repeatly.

6 Degree Distribution

6.1 Pseudocode

```
{Taken from lecture slide, modified}
Compute the degree, deg(v), of each vertex, v.
If G is represented as an adjacency list,
    count the number of elements in vâĂŹs list.
Create an array H, of size n, and initialize each H[i] = 0.
For each vertex, v
    increment H[deg(v)].
```

6.2 Description

Simply loop through all the nodes and get their degree. It generates a map that maps degree to number of nodes that have it.

7 Plots

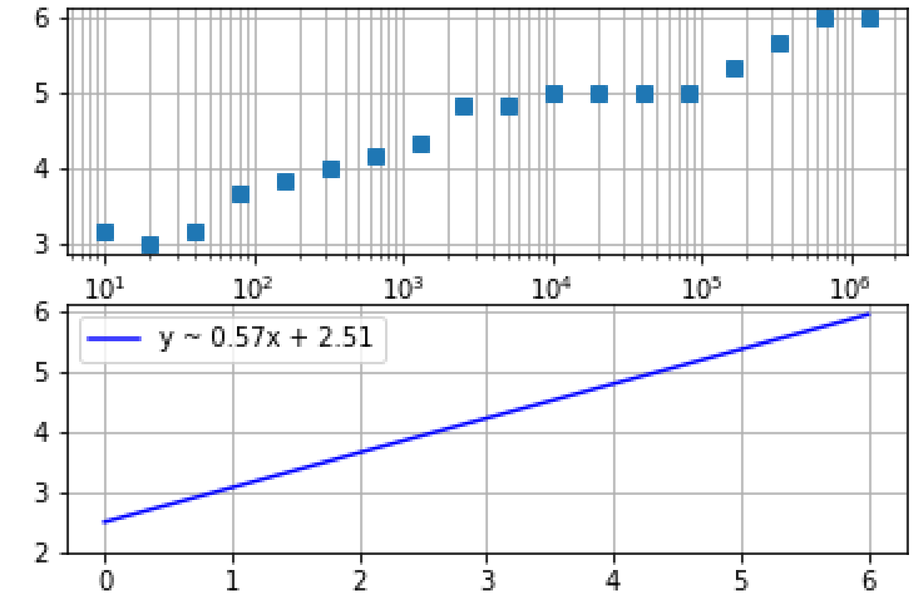


Figure 1: Diameter-n plot

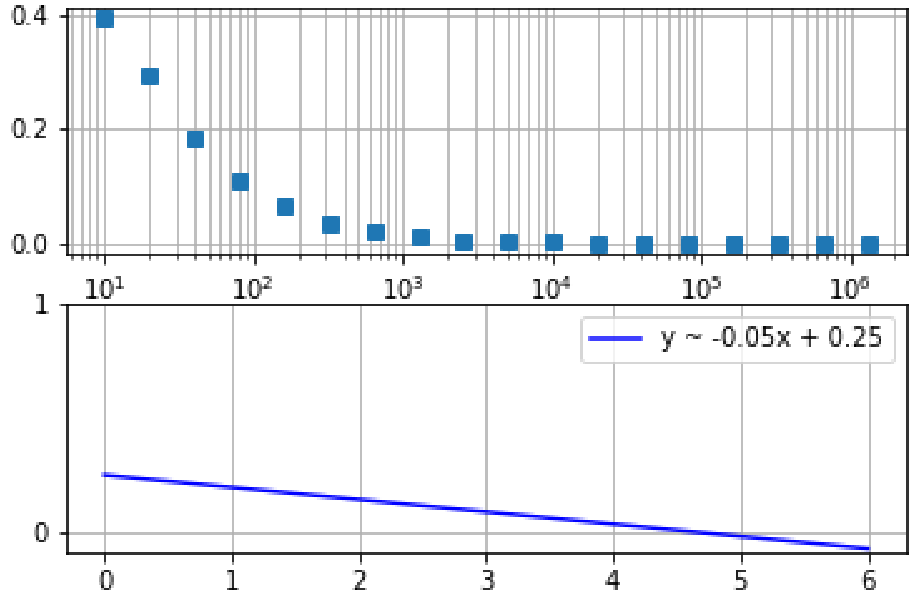


Figure 2: Cluster Coefficient-n plot

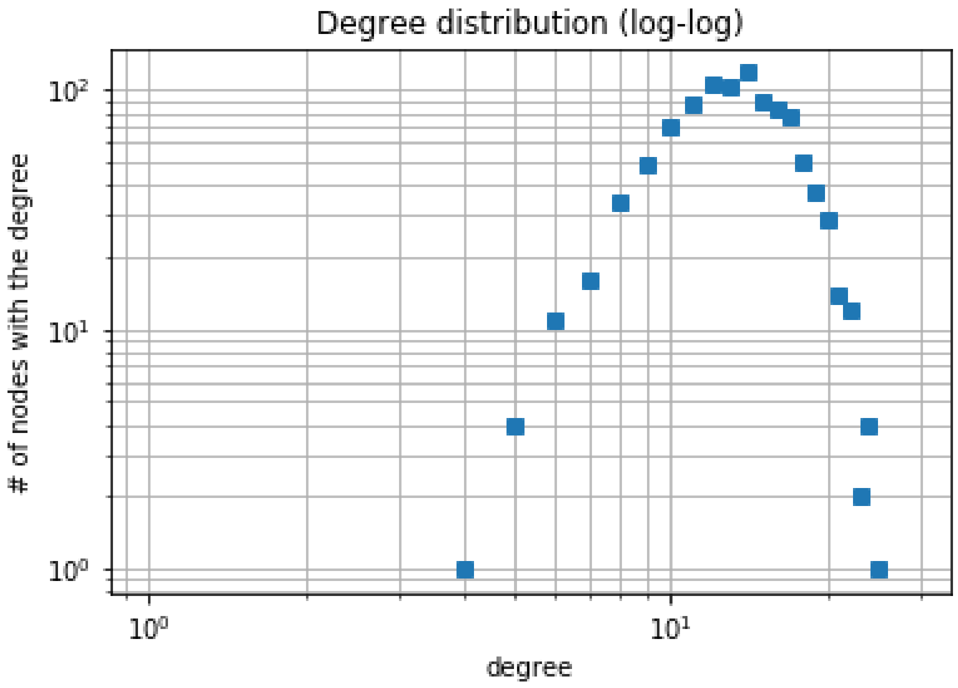


Figure 3: loglog Degree distribution w/ 1000 nodes

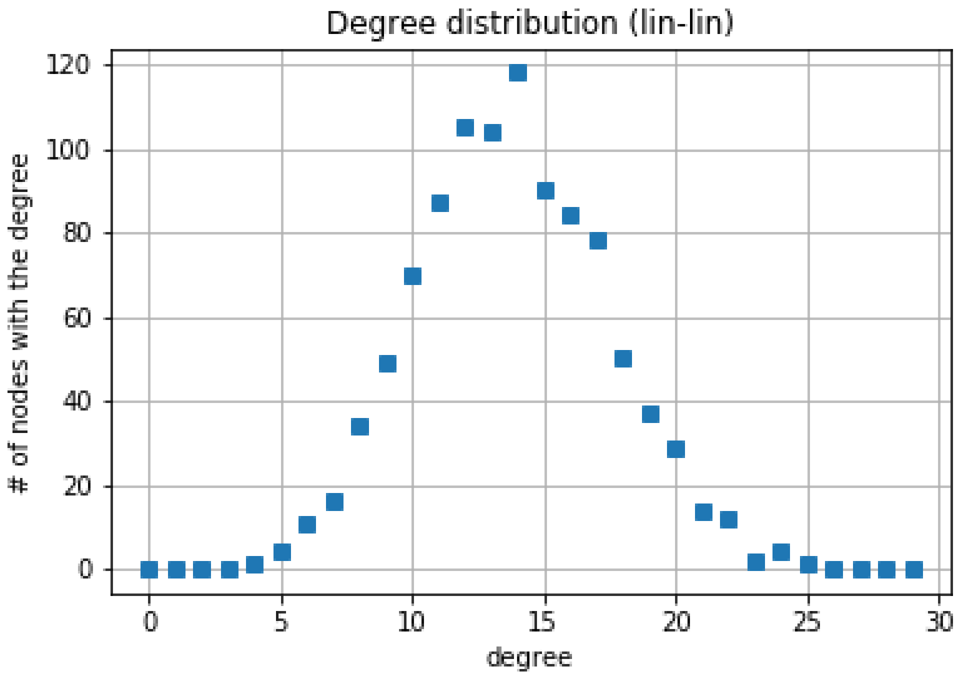


Figure 4: Degree distribution w/ 1000 nodes

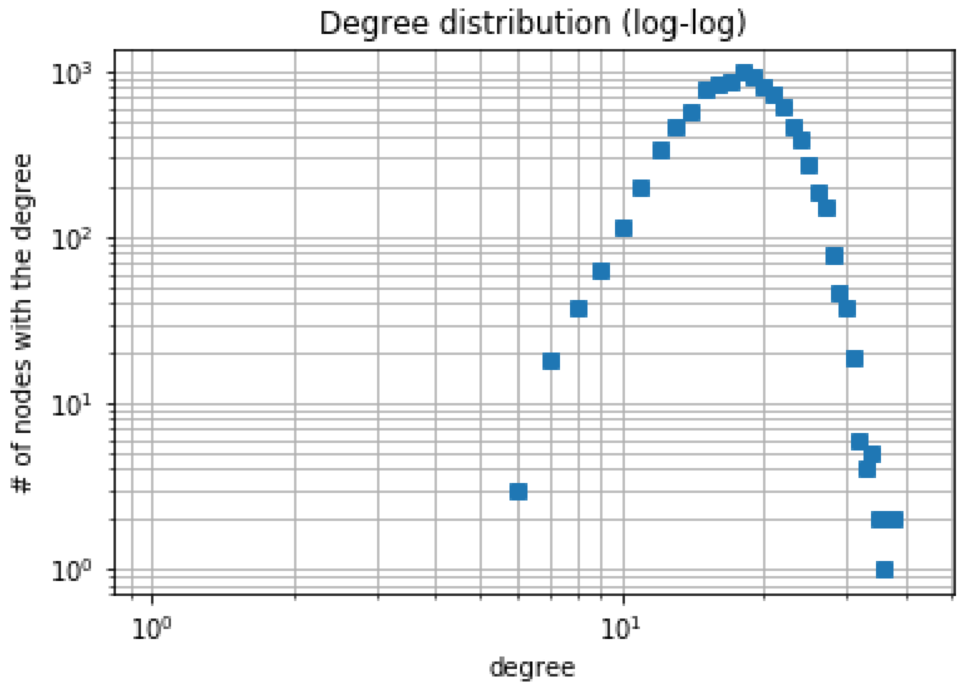


Figure 5: loglog Degree distribution w/ 10000 nodes

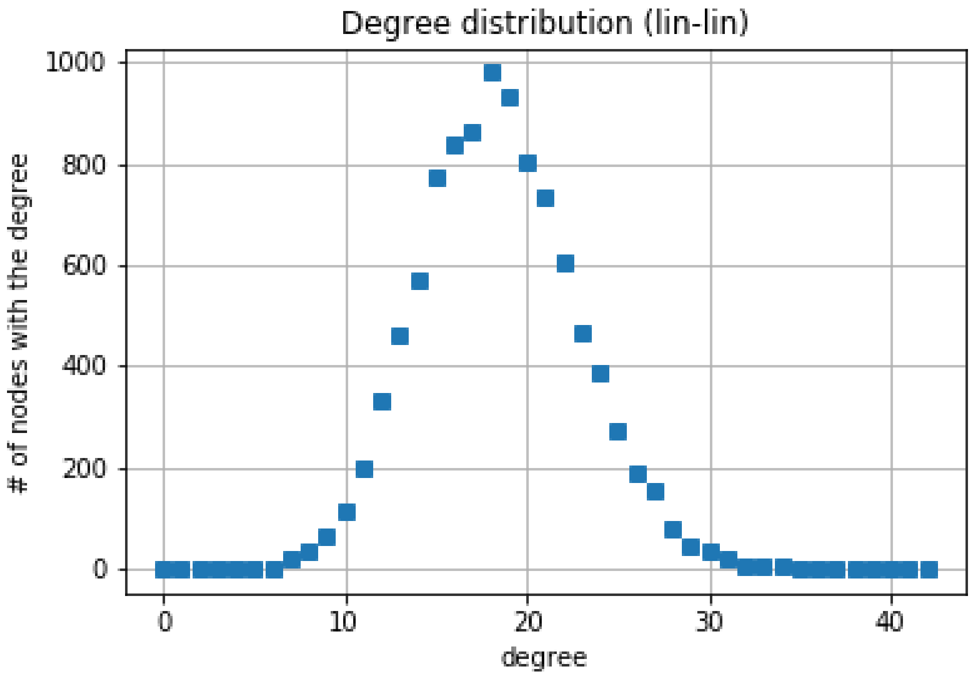


Figure 6: Degree distribution w/ 10000 nodes

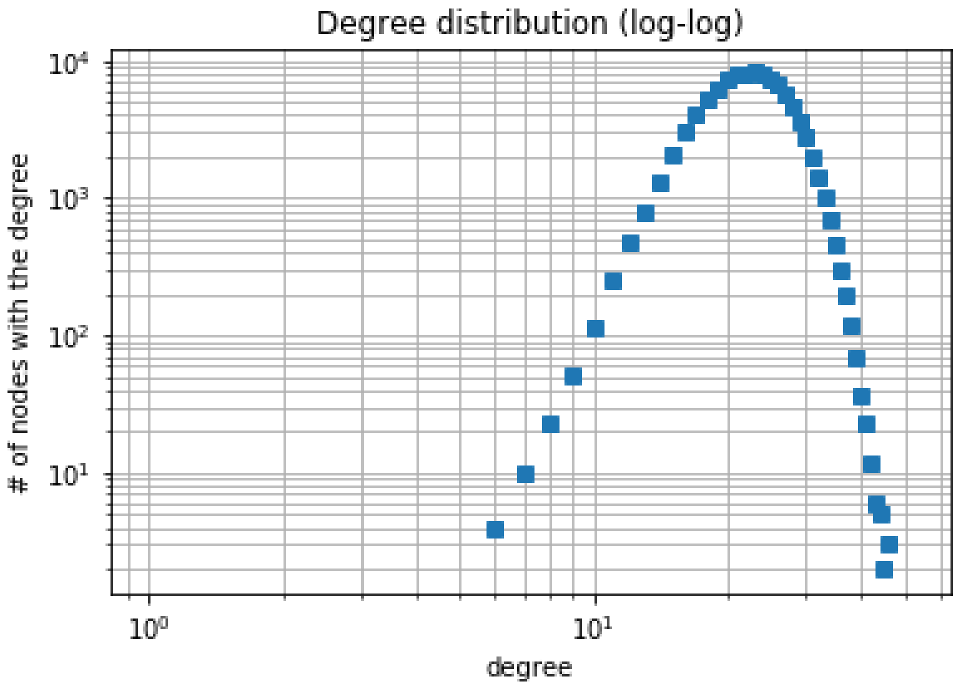


Figure 7: loglog Degree distribution w/ 100000 nodes

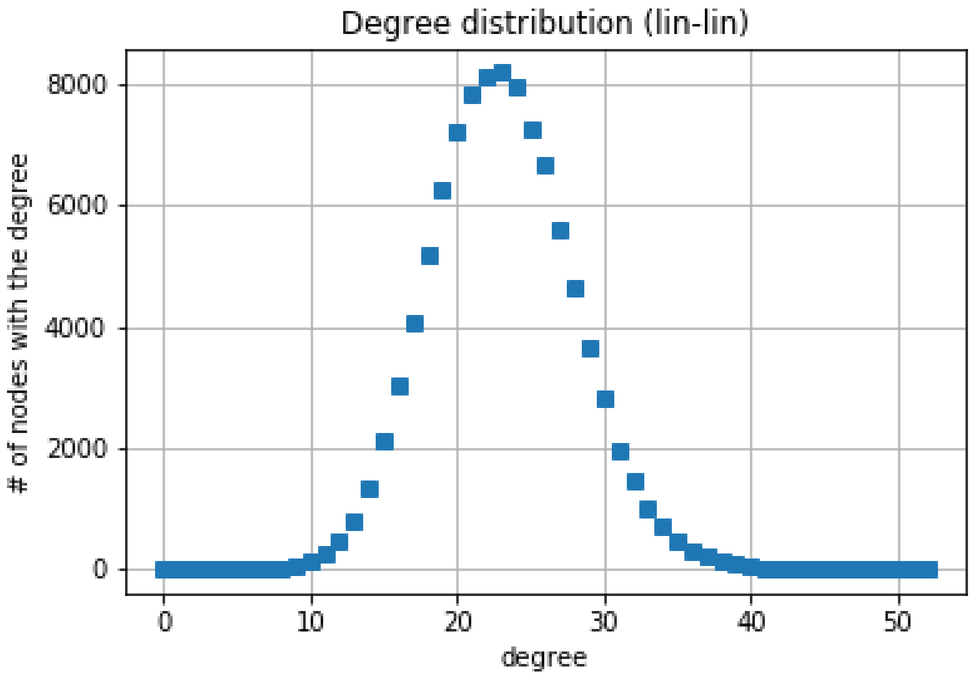


Figure 8: Degree distribution w/ 100000 nodes

7.1 Analysis

According to the plot of diameter growth, the approx. slope of the growth on lin-log scale is 0.57. The positive correlation tells us the **diameter of E-R network grows as a function of n.** However, since the function log(n) have a slope of 1 on lin-log. Since 0.57 is less than 1, **diameter grows slower than log(n).**
For cluster coefficient, the outcome is exact opposite. From the plot and the slope we get which is -0.05, we can conclude that **they decrease as function of n.**
As for degree distribution, from all three plots on linear scale, it is obvious that they don't have a heavy-tail. So **the degree distribution does not have a power law.**

8 Conclusion

The project gives a insight into the properties of Erdo-Renyi model. From the tests, some of its natures become prominent: In most cases it gives connected graph, or very few components. Also its diameter is small. The generated graph with given probability don't mimic real world network well, being that it does not give high clustering and heavy-tail degree distribution.