

# “网络科学基础”-第六次上机报告

班级： 物联网 2303                      姓名： 邱佳亮                      学号： 3230611072

上机日期： 2024.12.6, 第十四周周五下七八节课

2024 秋-网络科学基础（物联网 23）-第五次上机报告提交

## 一、上机题目

BA 无标度网络模型生成和图形界面综合编程

## 二、上机目的

1. 完成 BA 网络的生成，多次观察得到的 BA 网络图形。
2. 将之前若干次上机任务（至少随机网络模型、小世界模型、PR 值计算）功能代码集成到 Matlab 的 GUI 界面里，能根据界面的（参数）选择以完成对应的功能。

## 三、功能描述、上机程序（含必要的注释）、上机调试运行结果

1. 完成 BA 网络的生成，多次观察得到的 BA 网络图形

代码如下：

```
1.  clc;clear;
2.  m0=input("未增长前网络节点个数 m0:");
3.  m=input("引入新节点时新生成的边数 m:");
4.  N=input("增长后网络节点总数 N:");
5.
6.  disp("m0 节点连接情况:1 表示都是孤立点;2 表示构成完全图;3 表示随机连接");
7.  se=input("选择初始网络情况:");
8.  if m>m0
9.      disp("参数不合法");return;
10. end
11. x=100*rand(1,m0); %构造初始画图 m0 个节点
12. y=100*rand(1,m0);
13. if se==1
14.     A=zeros(m0);
15. elseif se==2
16.     A=ones(m0);
17.     A(1:m0+1:m0^2)=0; %对角线元素置 0
18.     %A(eye(size(A)))=0;
19. else
20.     A=zeros(m0);
21.     B=rand(m0);
```

```

22.     B=tril(B); %截取下三角元素
23.     A(B<=0.1)=1; %概率 0.1 进行连边
24.     A=A+A'; %构造邻接矩阵
25. end
26. for k=m0+1:N
27.     x(k)=100*rand; %生成画图坐标
28.     y(k)=100*rand;
29.     p=(sum(A)+1)/sum(sum(A)+1); %计算节点连接概率
30.     pp=cumsum(p); %求累计分布
31.     A(k,k)=0; %邻接矩阵扩充维数
32.     ind=[]; %初始集合
33.     while length(ind)<m
34.         jj=find(pp>rand); %赌轮法选择连边节点编号
35.         jj=jj(1);
36.         ind=union(ind,jj) %使用 union 保证节点不重复
37.     end
38.     A(k,ind)=1;
39.     A(ind,k)=1;%新的邻接矩阵
40. end
41. plot(x,y,'ro','MarkerEdgeColor','g','MarkerFaceColor','r','MarkerSize',8)
42. hold on;
43. A=tril(A);
44. [i,j]=find(A); %找邻接矩阵下三角元素的非零元素
45. for k=1:length(i)
46.     plot([x(i(k)),x(j(k))],[y(i(k)),y(j(k))],'LineWidth',1.2)
47.     filename=sprintf('plot_%d.png',k);
48.     saveas(gcf,filename);
49. end
50. deg=sum(A) %计算各节点的度
51. ave_degree=sum(deg)/N %计算平均度
52. figure,bar([1:N],deg); %画各节点度柱状图
53. title('网络图各节点度大小');
54. xlabel('$v_{i}$','Interpreter','latex');
55. ylabel('$k$','Interpreter','latex');
56. degrange=minmax(deg); %求度的取值范围
57. pinshu=hist(deg,[degrange(1):degrange(2)]); %求度取值的频数
58. df=pinshu/N; %度的频率分布
59. figure,bar([degrange(1):degrange(2)],df,'r'); %画度分布柱状图
60. title('网络图的度分布');
61. xlabel('$k$','Interpreter','latex');
62. ylabel('$P$','Interpreter','latex');
63. Matlab_to_Pajek(A,2)

```

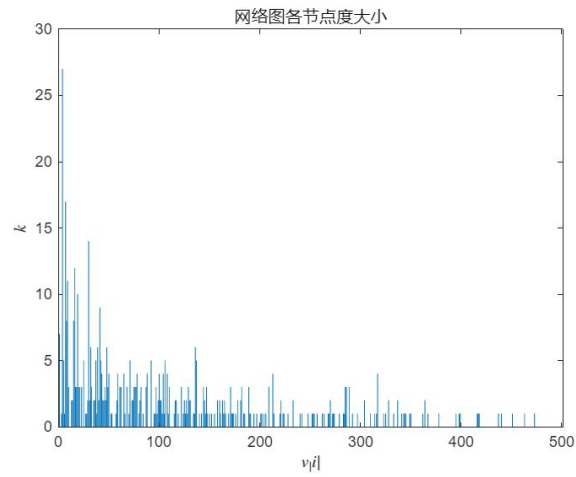


图 1 网络节点度的大小

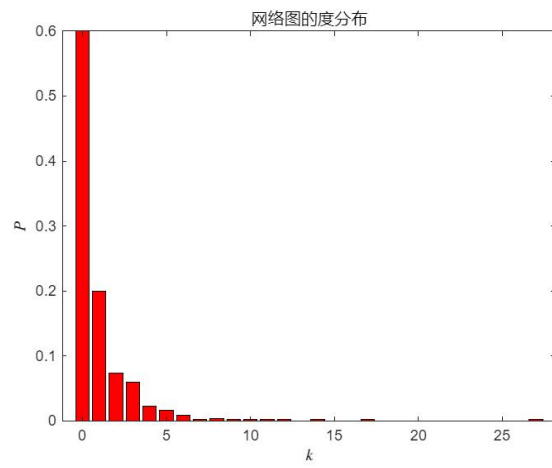


图 2 网络度分布

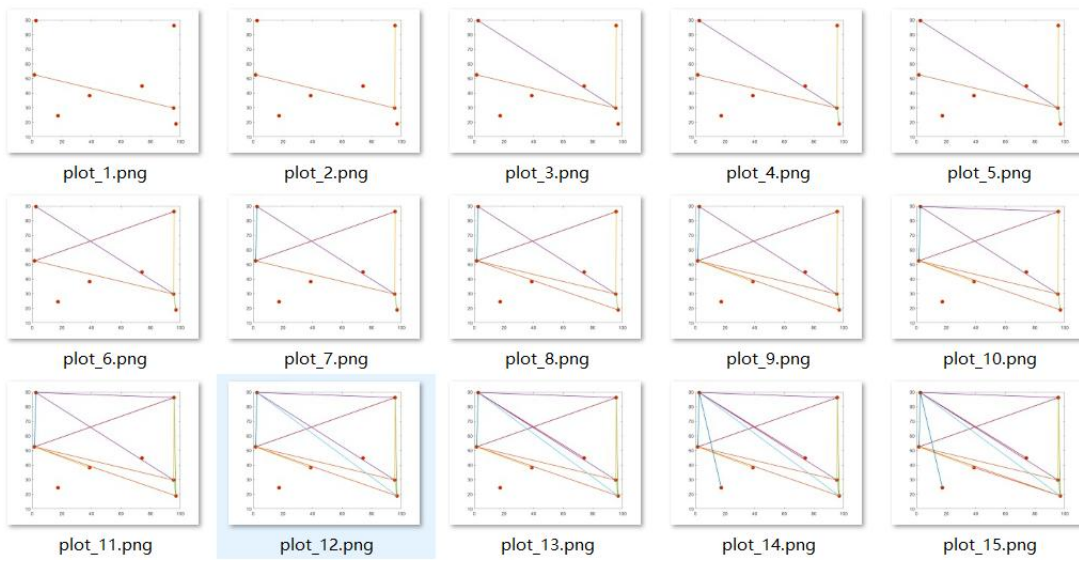


图 3 网络生成的中间过程

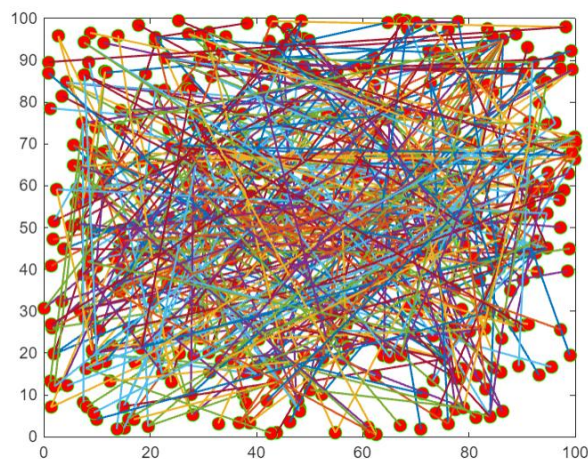


图 4 网络形态

2. 将之前若干次上机任务（至少随机网络模型、小世界模型、PR 值计算）功能代码集成到 Matlab 的 GUI 界面里，能根据界面的（参数）选择以完成对应的功能

在 GUIDE 中添加相应的按钮和弹出式菜单，设置两个绘图区已实现不同功能：

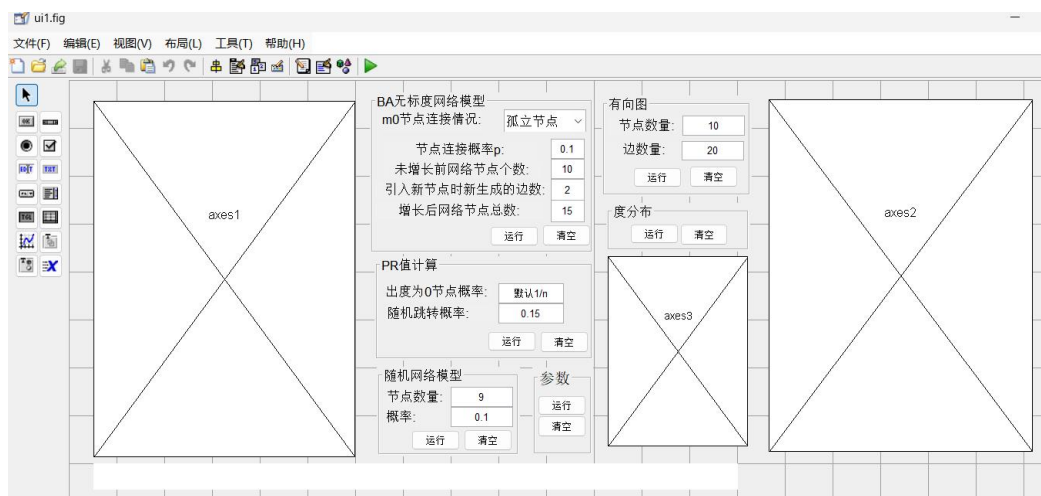


图 5 UI 设计

初始化函数：

```
1. function ui1_OpeningFcn(hObject, eventdata, handles, varargin)
2.     axes(handles.axes1); %清空绘图区
3.     cla reset;
4.     axes(handles.axes2);
5.     cla reset;
6.     handles.m0=10; %设置默认参数
7.     handles.m=2;
8.     handles.N=15;
```

```

9.     handles.mode=1;
10.    handles.p=0.1;
11.    handles.random_p=0.15;
12.    handles.zero_p=0;
13.    handles.num=9;
14.    handles.output = hObject;
15.    guidata(hObject, handles); %保存参数

```

选择 BA 无标度网络 m0 节点连接情况的弹出菜单的回调函数：

```

1.  function popupmenu1_Callback(hObject, eventdata, handles)
2.      str=get(hObject,'String'); % 获取当前对象
3.      val=get(hObject,'Value'); % 获取当前对象的值
4.      switch str{val} % 根据用户选择的下拉菜单项的索引，使用 switch 语句
        来决定执行哪个 case 分支
5.          case '孤立节点'
6.              handles.mode=1;
7.          case '构成完全图'
8.              handles.mode=2;
9.          case '随机连接'
10.             handles.mode=3;
11.      end
12.      guidata(hObject,handles) %保存

```

绘制 BA 无标度网络模型的按钮的回调函数：

```

1.  function pushbutton2_Callback(hObject, eventdata, handles)
2.      axes(handles.axes1); %清空绘图区
3.      cla reset;
4.      m0=handles.m0; %获取参数
5.      m=handles.m;
6.      N=handles.N;
7.      se=handles.mode;
8.      if m>m0
9.          disp("参数不合法");return;
10.     end
11.     x=100*rand(1,m0); %构造初始画图 m0 个节点
12.     y=100*rand(1,m0);
13.     if se==1
14.         A=zeros(m0);
15.     elseif se==2
16.         A=ones(m0);
17.         A(1:m0+1:m0^2)=0; %对角线元素置 0
18.     else
19.         A=zeros(m0);
20.         B=rand(m0);
21.         B=tril(B); %截取下三角元素

```

```

22.         A(B<=0.1)=1; %概率 0.1 进行连边
23.         A=A+A'; %构造邻接矩阵
24.     end
25.     for k=m0+1:N
26.         x(k)=100*rand; %生成画图坐标
27.         y(k)=100*rand;
28.         p=(sum(A)+1)/sum(sum(A)+1); %计算节点连接概率
29.         pp=cumsum(p); %求累计分布
30.         A(k,k)=0; %邻接矩阵扩充维数
31.         ind=[]; %初始集合
32.         while length(ind)<m
33.             jj=find(pp>rand); %赌轮法选择连边节点编号
34.             jj=jj(1);
35.             ind=union(ind,jj); %使用 union 保证节点不重复
36.         end
37.         A(k,ind)=1;
38.         A(ind,k)=1;%新的邻接矩阵
39.     end
40.     plot(handles.axes1,x,y,'ro','MarkerEdgeColor','g','MarkerFace
        Color','r','MarkerSize',8);
41.     hold on;
42.     handles.martix=A;
43.     A=tril(A);
44.     [i,j]=find(A); %找邻接矩阵下三角元素的非零元素
45.     for k=1:length(i)
46.         plot(handles.axes1,[x(i(k)),x(j(k))],[y(i(k)),y(j(k))],'L
            ineWidth',1.2)
47.     end
48.     guidata(hObject, handles);

```

输入 BA 无标度网络参数窗口的回调函数：

```

1.     function edit2_Callback(hObject, eventdata, handles)
2.         str=get(hObject,'String'); % 获取当前对象
3.         val=get(hObject,'Value'); % 获取当前对象的值
4.         handles.m0=str2double(val);
5.     function edit3_Callback(hObject, eventdata, handles)
6.         str=get(hObject,'String'); % 获取当前对象
7.         val=get(hObject,'Value'); % 获取当前对象的值
8.         handles.m=str2double(val);
9.     function edit4_Callback(hObject, eventdata, handles)
10.        str=get(hObject,'String'); % 获取当前对象
11.        val=get(hObject,'Value'); % 获取当前对象的值
12.        handles.N=str2double(val);

```

清空绘图区按钮的回调函数：

```

1. function pushbutton3_Callback(hObject, eventdata, handles)
2.     axes(handles.axes1); %清空绘图区
3.     cla reset;

```

绘制 PageRank 柱状图按钮的回调函数：

```

1. function pushbutton4_Callback(hObject, eventdata, handles)
2.     axes(handles.axes2); %清空绘图区
3.     cla reset;
4.     random_p=handles.random_p; %获取参数
5.     zero_p=handles.zero_p;
6.     B=handles.martix;
7.     r=sum(B,2); %计算出度
8.     n=length(B);
9.     if zero_p==0
10.         zero_p=1/n;
11.     end
12.     A=zeros(n);
13.     for i=1:n
14.         for j=1:n
15.             if r(i)>0
16.                 A(i,j)=random_p/n+(1-random_p)*B(i,j)/r(i);%构造状态转移矩阵
17.             else
18.                 A(i,j)=zero_p;
19.             end
20.         end
21.     end
22.     [x,y]=eigs(A',1);%特征向量归一化
23.     x=x/sum(x);%和为1
24.     bar(handles.axes2,x);%绘制柱状图

```

绘制随机网络模型按钮的回调函数：

```

1. function pushbutton6_Callback(hObject, eventdata, handles)
2.     axes(handles.axes1);
3.     cla reset;
4.     n=handles.num; %节点数量
5.     t=0:2*pi/n:2*pi; %生成角度向量
6.     m=nchoosek(n,2); %生成边的数量
7.     x=cos(t); %计算节点坐标
8.     y=sin(t);
9.     %axis([-1.1,1.1,-1.1,1.1]) %设置坐标轴范围
10.    %plot(handles.axes1,x,y,'o','Color','k') %绘制初始节点
11.    hold on;
12.    z=rand(1,m); %生成随机数
13.    p=handles.p;

```

```

14.     ind1=(z<=p); %决定哪些边保留
15.     ind2=squareform(ind1); %将逻辑向量转换为邻接矩阵
16.     handles.martix=ind2;
17.     [i,j]=find(ind2);
18.     plot(handles.axes1,x,y,'o','Color','k') %绘制节点
19.     for k=1:length(i) %绘制每条边
20.         line(handles.axes1,[x(i(k)),x(j(k))],[y(i(k)),y(j(k))],'Color','k')
21.     end
22.     guidata(hObject, handles);

```

计算绘图区网络模型参数的回调函数：

```

1.     function pushbutton8_Callback(hObject, eventdata, handles)
2.         A=handles.martix;
3.         d=sum(A); %度
4.         ave_degree=mean(d); %平均度
5.         d=sum(A); %计算度
6.         M=sum(d)/2; %总边数
7.         [i,j]=find(triu(A));
8.         ki=d(i);kj=d(j);
9.         r=(ki*kj'/M-(sum(ki+kj)/2/M)^2)/(sum(ki.^2+kj.^2)/2/M-(sum(ki+kj)/2/M)^2);
10.
11.         A=graph(A);
12.         dist=distances(A);
13.         D=max(max(dist));
14.         Ldist=tril(dist);
15.         he=sum(nonzeros(Ldist));
16.         n=numnodes(A);
17.         L=he/nchoosek(n,2);
18.         text15_handle=findobj('Tag','text15');
19.         set(text15_handle,'String',[ '同配系数为:',num2str(r),' ', '图直径为:', num2str(D) ...
20.                                     , ' ', '平均路径长度
21.                                     为:',num2str(L),' ', '总边数为:',num2str(M) ...
                                     , ' ', '平均度
                                     为:',num2str(ave_degree)])

```

清空模型参数区的回调函数：



```

1. function pushbutton9_Callback(hObject, eventdata, handles)
2.     text15_handle=findobj('Tag','text15');
3.     set(text15_handle,'String','');

```

绘制有向图：

```

1. function pushbutton10_Callback(hObject, eventdata, handles)
2.     axes(handles.axes1);
3.     cla reset;
4.     numNode=handles.numNode;
5.     numEdge=handles.s;
6.     srcNodes = randi(numNode, 1, numEdge); % 源节点
7.     tarNodes = randi(numNode, 1, numEdge); % 目标节点
8.
9.     G = digraph(srcNodes, tarNodes);
10.    handles.martix = adjacency(G);
11.    plot(handles.axes1,G);
12.    guidata(hObject,handles) %保存

```

绘制度分布：

```

1. function pushbutton12_Callback(hObject, eventdata, handles)
2.     A=handles.martix;
3.     N=length(A);
4.     deg=sum(A); %计算各节点的度
5.     degrange=minmax(deg); %求度的取值范围
6.     pinshu=hist(deg,[degrange(1):degrange(2)]); %求度取值的频数
7.     df=pinshu/N; %度的频率分布
8.     bar(handles.axes3,[degrange(1):degrange(2)],df,'r'); %画度分布
    柱状图

```

设置  $m_0$  节点为孤立结点、节点连接概率为  $p$ 、未增长前网络节点个数为 10、生成的边数为 2，增长后网络节点总数为 200，构建 BA 无标度网络，并绘制度分布：

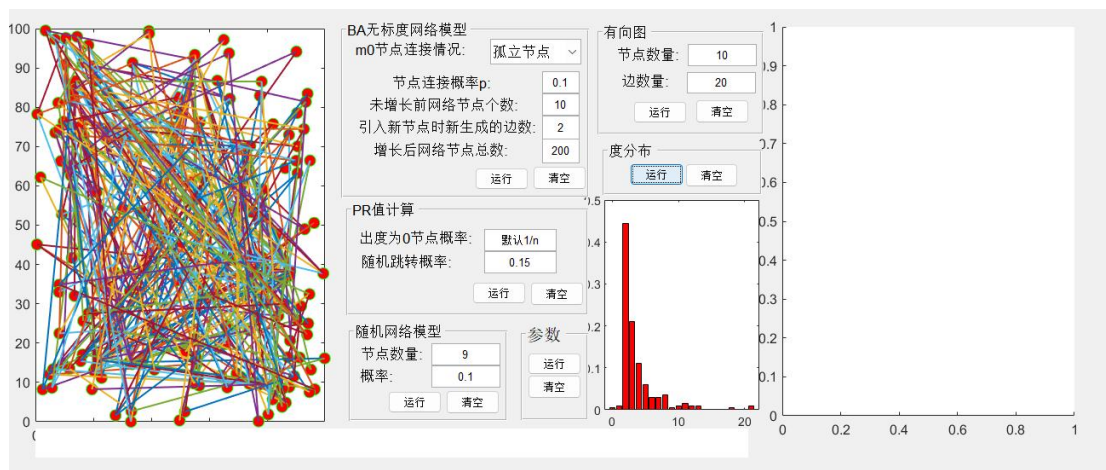


图 7 构建 BA 无标度网络

设置节点数量为 20，边数为 50，绘制有向图，并绘制 PageRank 值柱状图：

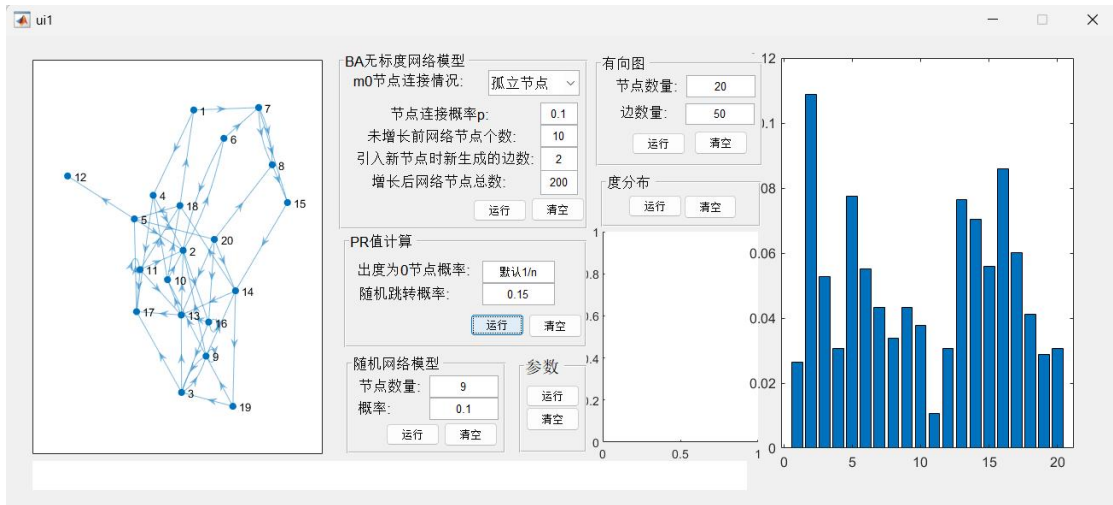


图 8 PageRank 值

设置节点数量为 50，连接概率为 0.1，构建随机网络模型，并计算 PageRank 值、绘制度分布并计算相关网络系数：

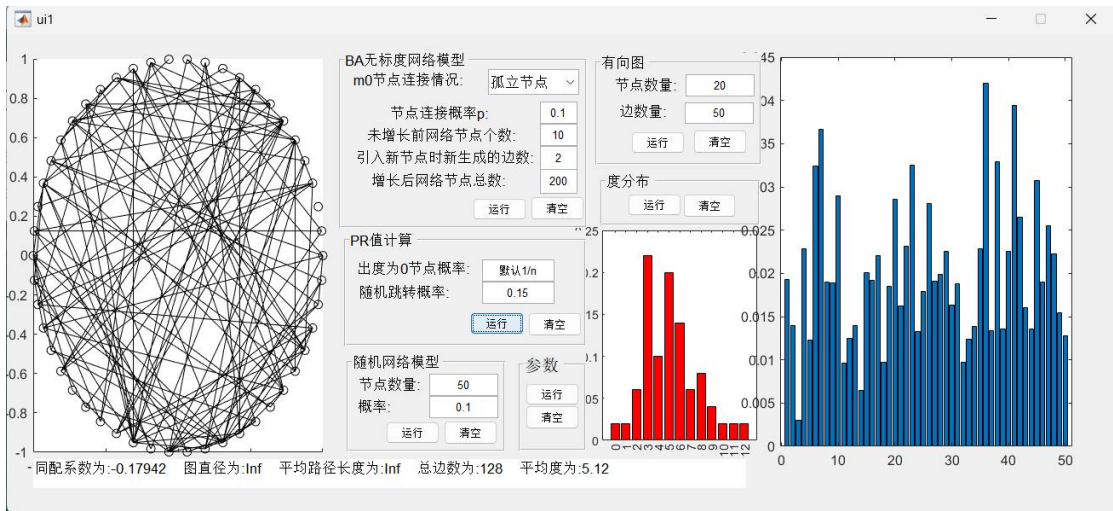


图 9 随机网络模型

#### 四、上机总结及感想

在完成这次“网络科学基础”的第六次上机报告后，我深刻体会到了网络科学的魅力和复杂性。通过亲手实现 BA 无标度网络模型的生成，我不仅加深了对无标度网络特性的理解，还锻炼了编程和问题解决能力。将随机网络模型、小世界模型和 PR 值计算等功能代码集成到 Matlab GUI 界面中，让我感受到了编程在解决实际问题中的应用价值。通过观察不同参数对网络结构的影响，我认识到了网络科学在模拟和分析复杂系统时的重要性。这次上机经历不仅提升了我的技术能

力，也激发了我对网络科学进一步探索的兴趣。

## 五、完整代码

```
1. function varargout = ui1(varargin)
2. gui_Singleton = 1;
3. gui_State = struct('gui_Name',       mfilename, ...
4.                   'gui_Singleton',   gui_Singleton, ...
5.                   'gui_OpeningFcn',   @ui1_OpeningFcn, ...
6.                   'gui_OutputFcn',   @ui1_OutputFcn, ...
7.                   'gui_LayoutFcn',    [] , ...
8.                   'gui_Callback',     []);
9. if nargin && ischar(varargin{1})
10.     gui_State.gui_Callback = str2func(varargin{1});
11. end
12.
13. if nargout
14.     [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
15. else
16.     gui_mainfcn(gui_State, varargin{:});
17. end
18. function ui1_OpeningFcn(hObject, eventdata, handles, varargin)
19.     axes(handles.axes1); %清空绘图区
20.     cla reset;
21.     axes(handles.axes2);
22.     cla reset;
23.     axes(handles.axes3);
24.     cla reset;
25.     handles.m0=10; %设置默认参数
26.     handles.m=2;
27.     handles.N=15;
28.     handles.mode=1;
29.     handles.p=0.1;
30.     handles.random_p=0.15;
31.     handles.zero_p=0;
32.     handles.num=9;
33.     handles.s=20;
34.     handles.numNode=10;
35.
36. handles.output = hObject;
37.
38. guidata(hObject, handles); %保存参数
39.
40. function varargout = ui1_OutputFcn(hObject, eventdata, handles)
```

```

41.
42. varargout{1} = handles.output;
43.
44. function popupmenu1_Callback(hObject, eventdata, handles)
45.     str=get(hObject,'String'); % 获取当前对象
46.     val=get(hObject,'Value'); % 获取当前对象的值
47.     switch str{val} % 根据用户选择的下拉菜单项的索引, 使用 switch 语句来决定执行哪个 case 分支
48.         case '孤立节点'
49.             handles.mode=1;
50.         case '构成完全图'
51.             handles.mode=2;
52.         case '随机连接'
53.             handles.mode=3;
54.     end
55.     guidata(hObject,handles) %保存
56. function popupmenu1_CreateFcn(hObject, eventdata, handles)
57. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
58.     set(hObject,'BackgroundColor','white');
59. end
60.
61. function edit1_Callback(hObject, eventdata, handles)
62.     str=get(hObject,'String'); % 获取当前对象
63.     val=get(hObject,'Value'); % 获取当前对象的值
64.     handles.p=str2double(str);
65.
66. function edit1_CreateFcn(hObject, eventdata, handles)
67. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
68.     set(hObject,'BackgroundColor','white');
69. end
70.
71. function pushbutton2_Callback(hObject, eventdata, handles)
72.     axes(handles.axes1); %清空绘图区
73.     cla reset;
74.     m0=handles.m0; %获取参数
75.     m=handles.m;
76.     N=handles.N;
77.     se=handles.mode;
78.     if m>m0

```

```

79.         disp("参数不合法");return;
80.     end
81.     x=100*rand(1,m0); %构造初始画图 m0 个节点
82.     y=100*rand(1,m0);
83.     if se==1
84.         A=zeros(m0);
85.     elseif se==2
86.         A=ones(m0);
87.         A(1:m0+1:m0^2)=0; %对角线元素置 0
88.     else
89.         A=zeros(m0);
90.         B=rand(m0);
91.         B=tril(B); %截取下三角元素
92.         A(B<=0.1)=1; %概率 0.1 进行连边
93.         A=A+A'; %构造邻接矩阵
94.     end
95.     for k=m0+1:N
96.         x(k)=100*rand; %生成画图坐标
97.         y(k)=100*rand;
98.         p=(sum(A)+1)/sum(sum(A)+1); %计算节点连接概率
99.         pp=cumsum(p); %求累计分布
100.        A(k,k)=0; %邻接矩阵扩充维数
101.        ind=[]; %初始集合
102.        while length(ind)<m
103.            jj=find(pp>rand); %赌轮法选择连边节点编号
104.            jj=jj(1);
105.            ind=union(ind,jj); %使用 unicon 保证节点不重复
106.        end
107.        A(k,ind)=1;
108.        A(ind,k)=1;%新的邻接矩阵
109.    end
110.    plot(handles.axes1,x,y,'ro','MarkerEdgeColor','g','MarkerFace
Color','r','MarkerSize',8);
111.    hold on;
112.    handles.martix=A;
113.    A=tril(A);
114.    [i,j]=find(A); %找邻接矩阵下三角元素的非零元素
115.    for k=1:length(i)
116.        plot(handles.axes1,[x(i(k)),x(j(k))],[y(i(k)),y(j(k))],'Li
newWidth',1.2)
117.    end
118.    guidata(hObject, handles);
119. function edit2_Callback(hObject, eventdata, handles)
120.     str=get(hObject,'String'); % 获取当前对象

```

```

121.     val=get(hObject,'Value'); % 获取当前对象的值
122.     handles.m0=str2double(str);
123.     guidata(hObject, handles);
124.
125. function edit2_CreateFcn(hObject, eventdata, handles)
126. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultU
icontrolBackgroundColor'))
127.     set(hObject,'BackgroundColor','white');
128. end
129.
130.
131.
132. function edit3_Callback(hObject, eventdata, handles)
133.     str=get(hObject,'String'); % 获取当前对象
134.     val=get(hObject,'Value'); % 获取当前对象的值
135.     handles.m=str2double(str);
136.     guidata(hObject, handles);
137. function edit3_CreateFcn(hObject, eventdata, handles)
138.
139. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultU
icontrolBackgroundColor'))
140.     set(hObject,'BackgroundColor','white');
141. end
142.
143. function edit4_Callback(hObject, eventdata, handles)
144.     str=get(hObject,'String'); % 获取当前对象
145.     val=get(hObject,'Value'); % 获取当前对象的值
146.     handles.N=str2double(str);
147.     guidata(hObject, handles);
148.
149. function edit4_CreateFcn(hObject, eventdata, handles)
150. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultU
icontrolBackgroundColor'))
151.     set(hObject,'BackgroundColor','white');
152. end
153.
154. function pushbutton3_Callback(hObject, eventdata, handles)
155.     axes(handles.axes1);
156.     cla reset;
157.
158. function pushbutton4_Callback(hObject, eventdata, handles)
159.     axes(handles.axes2); %清空绘图区
160.     cla reset;
161.     random_p=handles.random_p; %获取参数

```

```

162.     zero_p=handles.zero_p;
163.     B=handles.martix;
164.     r=sum(B,2); %计算出度
165.     n=length(B);
166.     if zero_p==0
167.         zero_p=1/n;
168.     end
169.     A=zeros(n);
170.     for i=1:n
171.         for j=1:n
172.             if r(i)>0
173.                 A(i,j)=random_p/n+(1-random_p)*B(i,j)/r(i);%构造状
态转移矩阵
174.             else
175.                 A(i,j)=zero_p;
176.             end
177.         end
178.     end
179.     [x,y]=eigs(A',[1]);%特征向量归一化
180.     x=x/sum(x);%和为 1
181.     bar(handles.axes2,x);%绘制柱状图
182.
183. function pushbutton5_Callback(hObject, eventdata, handles)
184.     axes(handles.axes2);
185.     cla reset;
186.
187. function edit5_Callback(hObject, eventdata, handles)
188.     str=get(hObject,'String'); % 获取当前对象
189.     val=get(hObject,'Value'); % 获取当前对象的值
190.     handles.zero_p=str2double(str);
191.     guidata(hObject, handles);
192. function edit5_CreateFcn(hObject, eventdata, handles)
193. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUi
controlBackgroundColor'))
194.     set(hObject,'BackgroundColor','white');
195. end
196.
197. function edit6_Callback(hObject, eventdata, handles)
198.     str=get(hObject,'String'); % 获取当前对象
199.     val=get(hObject,'Value'); % 获取当前对象的值
200.     handles.random_p=str2double(str);

```

```

201.     guidata(hObject, handles);
202.
203. function edit6_CreateFcn(hObject, eventdata, handles)
204. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUi
controlBackgroundColor'))
205.     set(hObject,'BackgroundColor','white');
206. end
207.
208. function edit8_Callback(hObject, eventdata, handles)
209.     str=get(hObject,'String'); % 获取当前对象
210.     val=get(hObject,'Value'); % 获取当前对象的值
211.     handles.num=str2double(str);
212.     guidata(hObject, handles);
213.
214. function edit8_CreateFcn(hObject, eventdata, handles)
215. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUi
controlBackgroundColor'))
216.     set(hObject,'BackgroundColor','white');
217. end
218.
219. function edit9_Callback(hObject, eventdata, handles)
220.     str=get(hObject,'String'); % 获取当前对象
221.     val=get(hObject,'Value'); % 获取当前对象的值
222.     handles.p=str2double(str);
223.     guidata(hObject, handles);
224.
225. function edit9_CreateFcn(hObject, eventdata, handles)
226. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUi
controlBackgroundColor'))
227.     set(hObject,'BackgroundColor','white');
228.
229. function pushbutton6_Callback(hObject, eventdata, handles)
230.     axes(handles.axes1);
231.     cla reset;
232.     n=handles.num; %节点数量
233.     t=0:2*pi/n:2*pi; %生成角度向量
234.     m=nchoosek(n,2); %生成边的数量
235.     x=cos(t); %计算节点坐标
236.     y=sin(t);
237.     %axis([-1.1,1.1,-1.1,1.1]) %设置坐标轴范围
238.     %plot(handles.axes1,x,y,'o','Color','k') %绘制初始节点
239.     hold on;

```



```

240.     z=rand(1,m); %生成随机数
241.     p=handles.p;
242.     ind1=(z<=p); %决定哪些边保留
243.     ind2=squareform(ind1); %将逻辑向量转换为邻接矩阵
244.     handles.martix=ind2;
245.     [i,j]=find(ind2);
246.     plot(handles.axes1,x,y,'o','Color','k') %绘制节点
247.     for k=1:length(i) %绘制每条边
248.         line(handles.axes1,[x(i(k)),x(j(k))],[y(i(k)),y(j(k))],'Color','k')
249.     end
250.     guidata(hObject, handles);
251.
252. function pushbutton7_Callback(hObject, eventdata, handles)
253.     axes(handles.axes1);
254.     cla reset;
255.
256. function pushbutton8_Callback(hObject, eventdata, handles)
257.     A=handles.martix;
258.     d=sum(A); %度
259.     ave_degree=mean(d); %平均度
260.
261.     d=sum(A); %计算度
262.     M=sum(d)/2; %总边数
263.     [i,j]=find(triu(A));
264.     ki=d(i);kj=d(j);
265.     r=(ki*kj'/M-(sum(ki+kj)/2/M)^2)/(sum(ki.^2+kj.^2)/2/M-(sum(ki+kj)/2/M)^2);
266.     if isequal(A',A)
267.         A=graph(A);
268.     else
269.         A=digraph(A);
270.     end
271.     dist=distances(A);
272.     D=max(max(dist));
273.     Ldist=tril(dist);
274.     he=sum(nonzeros(Ldist));
275.     n=numnodes(A);
276.     L=he/nchoosek(n,2);
277.     text15_handle=findobj('Tag','text15');
278.     set(text15_handle,'String',['同配系数为:',num2str(r),' ','图直径为:', num2str(D) ...
279.                                ,',' , '平均路径长度
为:',num2str(L),' ','总边数为:',num2str(M) ...

```

```

280.                                     , '      ', '平均度
为:', num2str(ave_degree)])
281.
282. function pushbutton9_Callback(hObject, eventdata, handles)
283.     text15_handle=findobj('Tag','text15');
284.     set(text15_handle,'String','');
285.
286. function text15_ButtonDownFcn(hObject, eventdata, handles)
287.     str=get(hObject,'String'); % 获取当前对象
288.     val=get(hObject,'Value'); % 获取当前对象的值
289.     handles.N=str2double(str);
290.     guidata(hObject, handles);
291.
292.
293. function edit10_Callback(hObject, eventdata, handles)
294.     str=get(hObject,'String'); % 获取当前对象
295.     val=get(hObject,'Value'); % 获取当前对象的值
296.     handles.numNode=str2double(str);
297.     guidata(hObject, handles);
298.
299. function edit10_CreateFcn(hObject, eventdata, handles)
300. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUi
controlBackgroundColor'))
301.     set(hObject,'BackgroundColor','white');
302. end
303.
304. function pushbutton10_Callback(hObject, eventdata, handles)
305.     axes(handles.axes1);
306.     cla reset;
307.     numNode=handles.numNode;
308.     numEdge=handles.s;
309.     srcNodes = randi(numNode, 1, numEdge); % 源节点
310.     tarNodes = randi(numNode, 1, numEdge); % 目标节点
311.
312.     G = digraph(srcNodes, tarNodes);
313.     handles.martix = adjacency(G);
314.     plot(handles.axes1,G);
315.     guidata(hObject,handles) %保存
316.
317. function pushbutton11_Callback(hObject, eventdata, handles)
318.     axes(handles.axes1);
319.     cla reset;

```

```

320.
321. function edit11_Callback(hObject, eventdata, handles)
322.     str=get(hObject,'String'); % 获取当前对象
323.     val=get(hObject,'Value'); % 获取当前对象的值
324.     handles.s=str2double(str);
325.     guidata(hObject, handles);
326.
327. function edit11_CreateFcn(hObject, eventdata, handles)
328. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUi
controlBackgroundColor'))
329.     set(hObject,'BackgroundColor','white');
330. end
331.
332. function pushbutton12_Callback(hObject, eventdata, handles)
333.     A=handles.martix;
334.     N=length(A);
335.     deg=sum(A); %计算各节点的度
336.     degrange=minmax(deg); %求度的取值范围
337.     pinshu=hist(deg,[degrange(1):degrange(2)]); %求度取值的频数
338.     df=pinshu/N; %度的频率分布
339.     bar(handles.axes3,[degrange(1):degrange(2)],df,'r'); %画度分布柱
状图
340.
341. function pushbutton13_Callback(hObject, eventdata, handles)
342.     axes(handles.axes3);
343.     cla reset;

```