**“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

1. **上机题目**

BA 无标度网络模型生成和图形界面综合编程

1. **上机目的**
2. 完成 BA 网络的生成，多次观察得到的 BA 网络图形。
3. 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。
4. **功能描述、上机程序（含必要的注释）、上机调试运行结果**

### 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. 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");
6. se=input("选择初始网络情况:");
7. if m>m0
8. disp("参数不合法");return;
9. end
10. x=100\*rand(1,m0); %构造初始画图m0个节点
11. y=100\*rand(1,m0);
12. if se==1
13. A=zeros(m0);
14. elseif se==2
15. A=ones(m0);
16. A(1:m0+1:m0^2)=0; %对角线元素置0
17. %A(eye(size(A)))=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unicn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(x,y,'ro','MarkerEdgeColor','g','MarkerFaceColor','r','MarkerSize',8)
41. hold on;
42. A=tril(A);
43. [i,j]=find(A); %找邻接矩阵下三角元素的非零元素
44. for k=1:length(i)
45. plot([x(i(k)),x(j(k))],[y(i(k)),y(j(k))],'LineWidth',1.2)
46. filename=sprintf('plot\_%d.png',k);
47. saveas(gcf,filename);
48. end
49. deg=sum(A) %计算各节点的度
50. ave\_degree=sum(deg)/N %计算平均度
51. figure,bar([1:N],deg); %画各节点度柱状图
52. title('网络图各节点度大小');
53. xlabel('$v\_|i|$','Interpreter','latex');
54. ylabel('$k$','Interpreter','latex');
55. degrange=minmax(deg); %求度的取值范围
56. pinshu=hist(deg,[degrange(1):degrange(2)]); %求度取值的频数
57. df=pinshu/N; %度的频率分布
58. figure,bar([degrange(1):degrange(2)],df,'r'); %画度分布柱状图
59. title('网络图的度分布');
60. xlabel('$k$','Interpreter','latex');
61. ylabel('$P$','Interpreter','latex');
62. Matlab\_to\_Pajek(A,2)

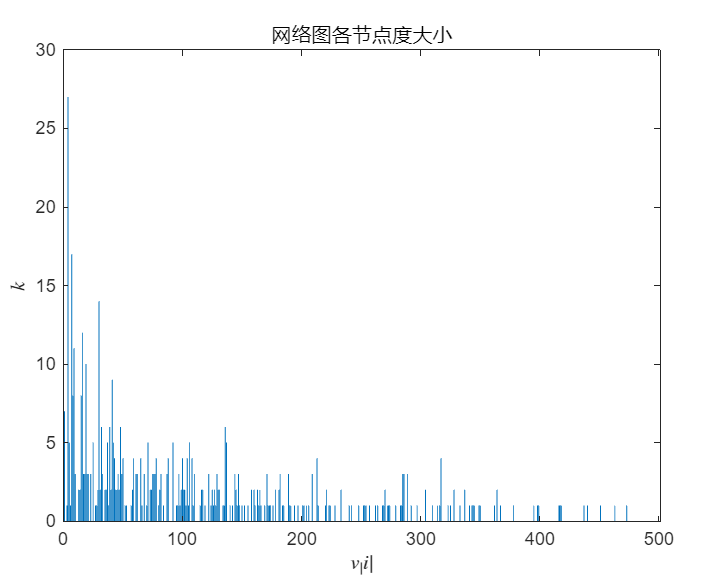


图 1 网络节点度的大小

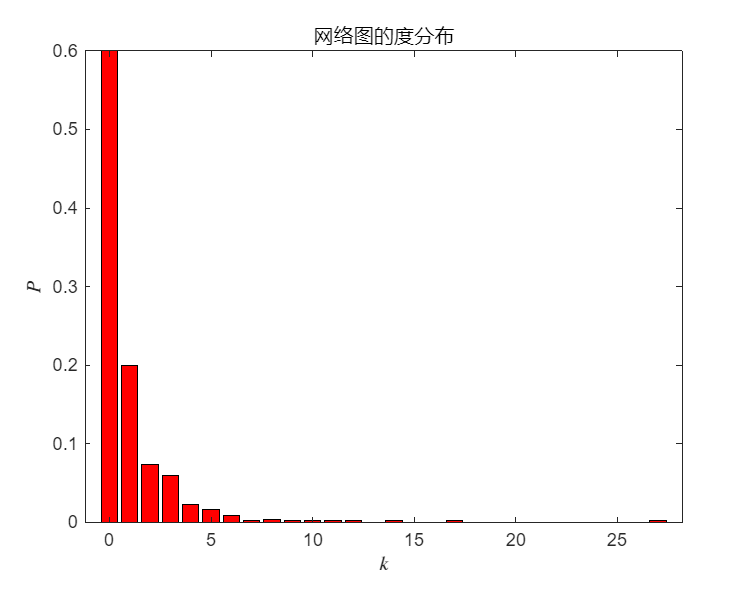


图 2 网络度分布

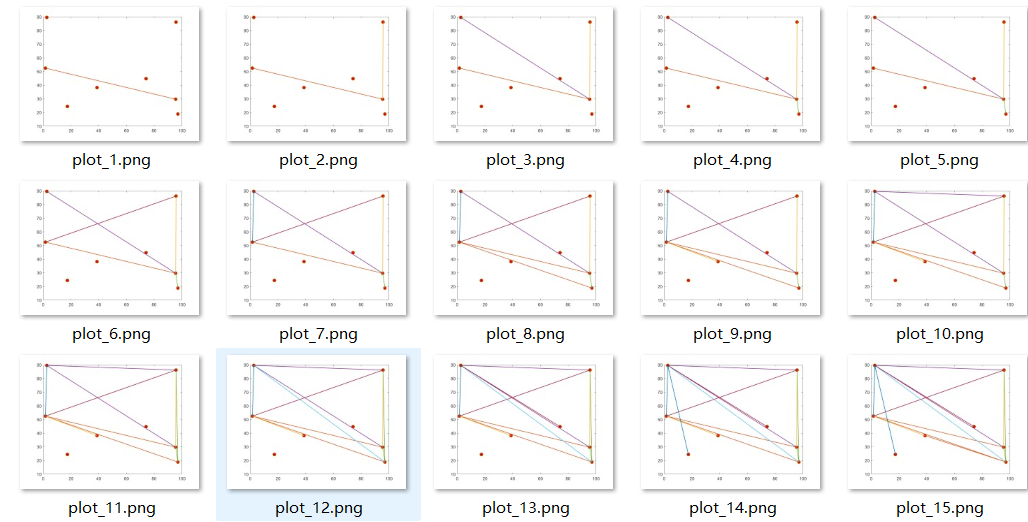


图 3 网络生成的中间过程

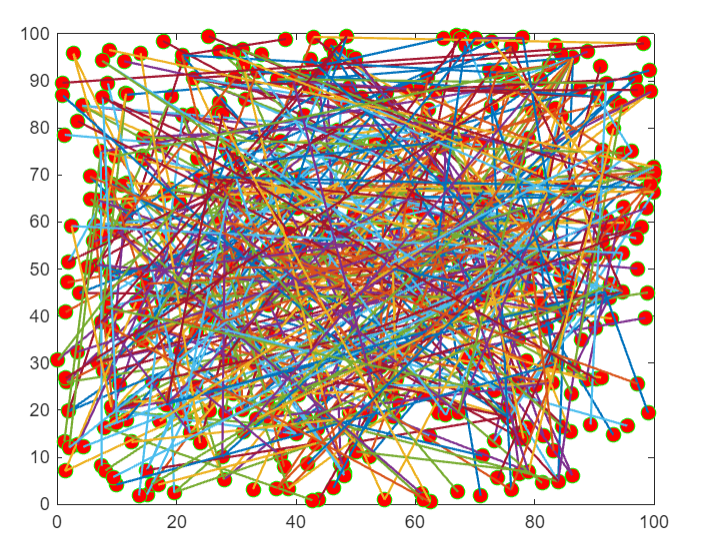


图 4 网络形态

### 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unicn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','MarkerFaceColor','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))],*'LineWidth',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);
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. ,'    ','平均路径长度为:',num2str(L),'    ','总边数为:',num2str(M) ...
21. ,'    ','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
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m0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：

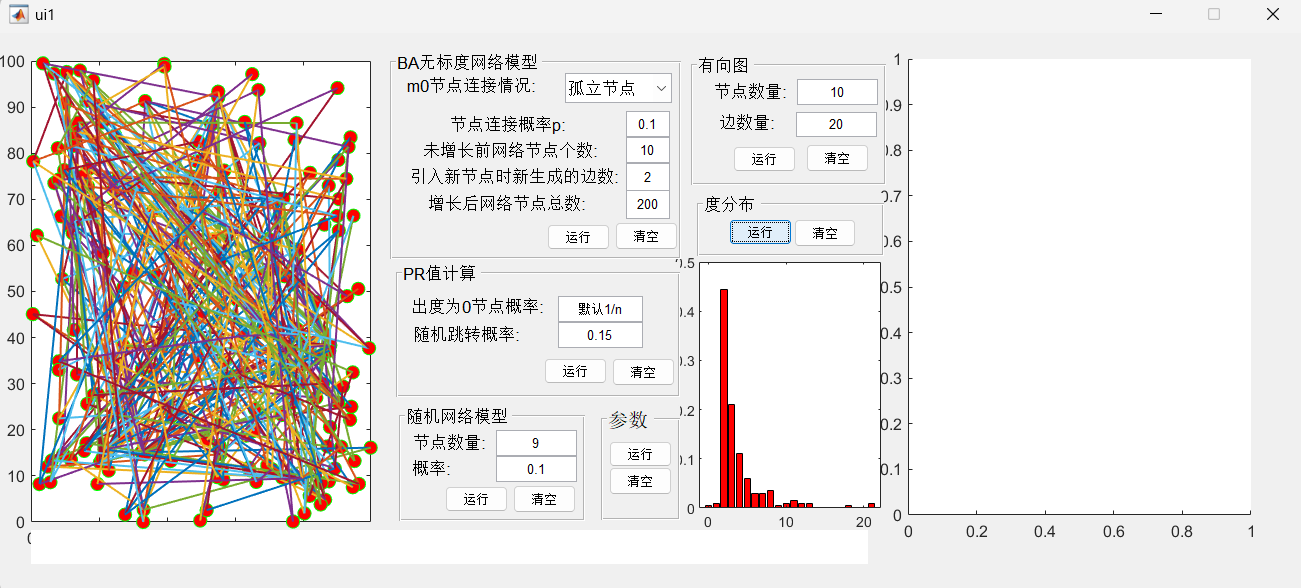


图 7 构建BA无标度网络

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：

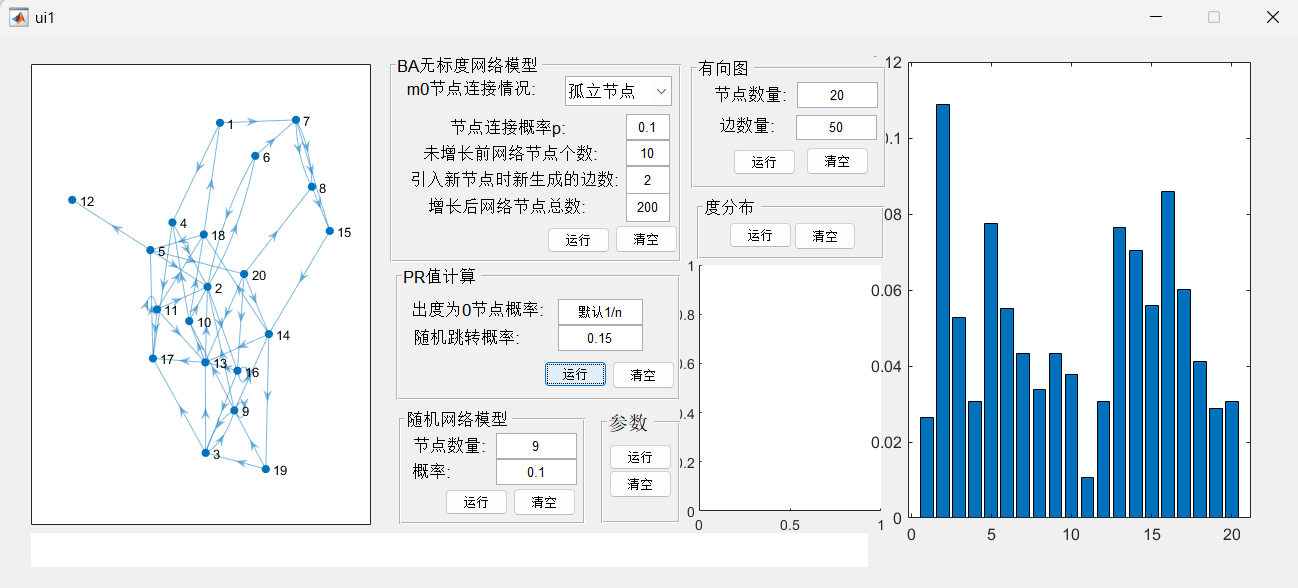


图 8 PageRank值

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：

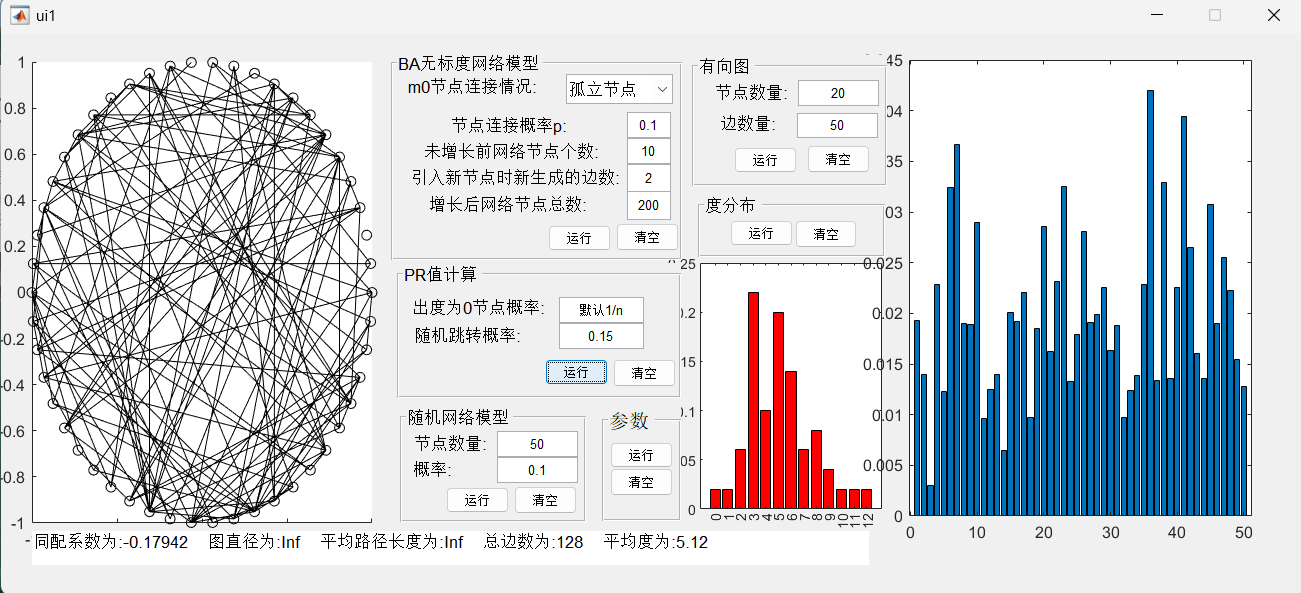


图 9 随机网络模型

1. **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。

1. **完整代码**
2. function varargout = ui1(varargin)
3. gui\_Singleton = 1;
4. gui\_State = struct('gui\_Name',       mfilename, ...
5. 'gui\_Singleton',  gui\_Singleton, ...
6. 'gui\_OpeningFcn', @ui1\_OpeningFcn, ...
7. 'gui\_OutputFcn',  @ui1\_OutputFcn, ...
8. 'gui\_LayoutFcn',  [] , ...
9. 'gui\_Callback',   []);
10. if nargin && ischar(varargin{1})
11. gui\_State.gui\_Callback = str2func(varargin{1});
12. end
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;
36. handles.output = hObject;
37. guidata(hObject, handles); %保存参数
38. function varargout = ui1\_OutputFcn(hObject, eventdata, handles)
39. varargout{1} = handles.output;
40. function popupmenu1\_Callback(hObject, eventdata, handles)
41. str=get(hObject,'String'); % 获取当前对象
42. val=get(hObject,'Value'); % 获取当前对象的值
43. 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
44. case '孤立节点'
45. handles.mode=1;
46. case '构成完全图'
47. handles.mode=2;
48. case '随机连接'
49. handles.mode=3;
50. end
51. guidata(hObject,handles) %保存
52. function popupmenu1\_CreateFcn(hObject, eventdata, handles)
53. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
54. set(hObject,'BackgroundColor','white');
55. end
56. function edit1\_Callback(hObject, eventdata, handles)
57. str=get(hObject,'String'); % 获取当前对象
58. val=get(hObject,'Value'); % 获取当前对象的值
59. handles.p=str2double(str);
60. function edit1\_CreateFcn(hObject, eventdata, handles)
61. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
62. set(hObject,'BackgroundColor','white');
63. end
64. function pushbutton2\_Callback(hObject, eventdata, handles)
65. axes(handles.axes1); %清空绘图区
66. cla reset;
67. m0=handles.m0; %获取参数
68. m=handles.m;
69. N=handles.N;
70. se=handles.mode;
71. if m>m0
72. disp("参数不合法");return;
73. end
74. x=100\*rand(1,m0); %构造初始画图m0个节点
75. y=100\*rand(1,m0);
76. if se==1
77. A=zeros(m0);
78. elseif se==2
79. A=ones(m0);
80. A(1:m0+1:m0^2)=0; %对角线元素置0
81. else
82. A=zeros(m0);
83. B=rand(m0);
84. B=tril(B); %截取下三角元素
85. A(B<=0.1)=1; %概率0.1进行连边
86. A=A+A'; %构造邻接矩阵
87. end
88. for k=m0+1:N
89. x(k)=100\*rand; %生成画图坐标
90. y(k)=100\*rand;
91. p=(sum(A)+1)/sum(sum(A)+1); %计算节点连接概率
92. pp=cumsum(p); %求累计分布
93. A(k,k)=0; %邻接矩阵扩充维数
94. ind=[]; %初始集合
95. while length(ind)<m
96. jj=find(pp>rand); %赌轮法选择连边节点编号
97. jj=jj(1);
98. ind=union(ind,jj); %使用unicn保证节点不重复
99. end
100. A(k,ind)=1;
101. A(ind,k)=1;%新的邻接矩阵
102. end
103. plot(handles.axes1,x,y,'ro','MarkerEdgeColor','g','MarkerFaceColor','r','MarkerSize',8);
104. hold on;
105. handles.martix=A;
106. A=tril(A);
107. [i,j]=find(A); %找邻接矩阵下三角元素的非零元素
108. for k=1:length(i)
109. plot(handles.axes1,[x(i(k)),x(j(k))],[y(i(k)),y(j(k))],'LineWidth',1.2)
110. end
111. guidata(hObject, handles);
112. function edit2\_Callback(hObject, eventdata, handles)
113. str=get(hObject,'String'); % 获取当前对象
114. val=get(hObject,'Value'); % 获取当前对象的值
115. handles.m0=str2double(str);
116. guidata(hObject, handles);
117. function edit2\_CreateFcn(hObject, eventdata, handles)
118. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
119. set(hObject,'BackgroundColor','white');
120. end
121. function edit3\_Callback(hObject, eventdata, handles)
122. str=get(hObject,'String'); % 获取当前对象
123. val=get(hObject,'Value'); % 获取当前对象的值
124. handles.m=str2double(str);
125. guidata(hObject, handles);
126. function edit3\_CreateFcn(hObject, eventdata, handles)
127. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
128. set(hObject,'BackgroundColor','white');
129. end
130. function edit4\_Callback(hObject, eventdata, handles)
131. str=get(hObject,'String'); % 获取当前对象
132. val=get(hObject,'Value'); % 获取当前对象的值
133. handles.N=str2double(str);
134. guidata(hObject, handles);
135. function edit4\_CreateFcn(hObject, eventdata, handles)
136. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
137. set(hObject,'BackgroundColor','white');
138. end
139. function pushbutton3\_Callback(hObject, eventdata, handles)
140. axes(handles.axes1);
141. cla reset;
142. function pushbutton4\_Callback(hObject, eventdata, handles)
143. axes(handles.axes2); %清空绘图区
144. cla reset;
145. random\_p=handles.random\_p; %获取参数
146. zero\_p=handles.zero\_p;
147. B=handles.martix;
148. r=sum(B,2); %计算出度
149. n=length(B);
150. if zero\_p==0
151. zero\_p=1/n;
152. end
153. A=zeros(n);
154. for i=1:n
155. for j=1:n
156. if r(i)>0
157. A(i,j)=random\_p/n+(1-random\_p)\*B(i,j)/r(i);%构造状态转移矩阵
158. else
159. A(i,j)=zero\_p;
160. end
161. end
162. end
163. [x,y]=eigs(A',1);%特征向量归一化
164. x=x/sum(x);%和为1
165. bar(handles.axes2,x);%绘制柱状图
166. function pushbutton5\_Callback(hObject, eventdata, handles)
167. axes(handles.axes2);
168. cla reset;
169. function edit5\_Callback(hObject, eventdata, handles)
170. str=get(hObject,'String'); % 获取当前对象
171. val=get(hObject,'Value'); % 获取当前对象的值
172. handles.zero\_p=str2double(str);
173. guidata(hObject, handles);
174. function edit5\_CreateFcn(hObject, eventdata, handles)
175. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
176. set(hObject,'BackgroundColor','white');
177. end
178. function edit6\_Callback(hObject, eventdata, handles)
179. str=get(hObject,'String'); % 获取当前对象
180. val=get(hObject,'Value'); % 获取当前对象的值
181. handles.random\_p=str2double(str);
182. guidata(hObject, handles);
183. function edit6\_CreateFcn(hObject, eventdata, handles)
184. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
185. set(hObject,'BackgroundColor','white');
186. end
187. function edit8\_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.num=str2double(str);
191. guidata(hObject, handles);
192. function edit8\_CreateFcn(hObject, eventdata, handles)
193. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
194. set(hObject,'BackgroundColor','white');
195. end
196. function edit9\_Callback(hObject, eventdata, handles)
197. str=get(hObject,'String'); % 获取当前对象
198. val=get(hObject,'Value'); % 获取当前对象的值
199. handles.p=str2double(str);
200. guidata(hObject, handles);
201. function edit9\_CreateFcn(hObject, eventdata, handles)
202. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
203. set(hObject,'BackgroundColor','white');
204. function pushbutton6\_Callback(hObject, eventdata, handles)
205. axes(handles.axes1);
206. cla reset;
207. n=handles.num; %节点数量
208. t=0:2\*pi/n:2\*pi; %生成角度向量
209. m=nchoosek(n,2); %生成边的数量
210. x=cos(t); %计算节点坐标
211. y=sin(t);
212. %axis([-1.1,1.1,-1.1,1.1]) %设置坐标轴范围
213. %plot(handles.axes1,x,y,'o','Color','k') %绘制初始节点
214. hold on;
215. z=rand(1,m); %生成随机数
216. p=handles.p;
217. ind1=(z<=p); %决定哪些边保留
218. ind2=squareform(ind1); %将逻辑向量转换为邻接矩阵
219. handles.martix=ind2;
220. [i,j]=find(ind2);
221. plot(handles.axes1,x,y,'o','Color','k') %绘制节点
222. for k=1:length(i) %绘制每条边
223. line(handles.axes1,[x(i(k)),x(j(k))],[y(i(k)),y(j(k))],'Color','k')
224. end
225. guidata(hObject, handles);
226. function pushbutton7\_Callback(hObject, eventdata, handles)
227. axes(handles.axes1);
228. cla reset;
229. function pushbutton8\_Callback(hObject, eventdata, handles)
230. A=handles.martix;
231. d=sum(A); %度
232. ave\_degree=mean(d); %平均度
234. d=sum(A); %计算度
235. M=sum(d)/2; %总边数
236. [i,j]=find(triu(A));
237. ki=d(i);kj=d(j);
238. r=(ki\*kj'/M-(sum(ki+kj)/2/M)^2)/(sum(ki.^2+kj.^2)/2/M-(sum(ki+kj)/2/M)^2);
239. if isequal(A',A)
240. A=graph(A);
241. else
242. A=digraph(A);
243. end
244. dist=distances(A);
245. D=max(max(dist));
246. Ldist=tril(dist);
247. he=sum(nonzeros(Ldist));
248. n=numnodes(A);
249. L=he/nchoosek(n,2);
250. text15\_handle=findobj('Tag','text15');
251. set(text15\_handle,'String',['同配系数为:',num2str(r),'    ','图直径为:', num2str(D) ...
252. ,'    ','平均路径长度为:',num2str(L),'    ','总边数为:',num2str(M) ...
253. ,'    ','平均度为:',num2str(ave\_degree)])
254. function pushbutton9\_Callback(hObject, eventdata, handles)
255. text15\_handle=findobj('Tag','text15');
256. set(text15\_handle,'String','');
257. function text15\_ButtonDownFcn(hObject, eventdata, handles)
258. str=get(hObject,'String'); % 获取当前对象
259. val=get(hObject,'Value'); % 获取当前对象的值
260. handles.N=str2double(str);
261. guidata(hObject, handles);
262. function edit10\_Callback(hObject, eventdata, handles)
263. str=get(hObject,'String'); % 获取当前对象
264. val=get(hObject,'Value'); % 获取当前对象的值
265. handles.numNode=str2double(str);
266. guidata(hObject, handles);
267. function edit10\_CreateFcn(hObject, eventdata, handles)
268. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
269. set(hObject,'BackgroundColor','white');
270. end
271. function pushbutton10\_Callback(hObject, eventdata, handles)
272. axes(handles.axes1);
273. cla reset;
274. numNode=handles.numNode;
275. numEdge=handles.s;
276. srcNodes = randi(numNode, 1, numEdge); % 源节点
277. tarNodes = randi(numNode, 1, numEdge); % 目标节点
279. G = digraph(srcNodes, tarNodes);
280. handles.martix = adjacency(G);
281. plot(handles.axes1,G);
282. guidata(hObject,handles) %保存
283. function pushbutton11\_Callback(hObject, eventdata, handles)
284. axes(handles.axes1);
285. cla reset;
286. function edit11\_Callback(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.s=str2double(str);
290. guidata(hObject, handles);
292. function edit11\_CreateFcn(hObject, eventdata, handles)
293. if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
294. set(hObject,'BackgroundColor','white');
295. end
296. function pushbutton12\_Callback(hObject, eventdata, handles)
297. A=handles.martix;
298. N=length(A);
299. deg=sum(A); %计算各节点的度
300. degrange=minmax(deg); %求度的取值范围
301. pinshu=hist(deg,[degrange(1):degrange(2)]); %求度取值的频数
302. df=pinshu/N; %度的频率分布
303. bar(handles.axes3,[degrange(1):degrange(2)],df,'r'); %画度分布柱状图
304. function pushbutton13\_Callback(hObject, eventdata, handles)
305. axes(handles.axes3);
306. cla reset;