用程序生成WS模型，并且对其小世界特性进行计算。

WS模型从一个完全的规则网络出发，以一定的概率将网络中的连接打乱重连。构建一个含有N个点的环形最近邻耦合网络，每个节点都与它左右相邻的K个节点相连 。以概率p随机地重新连接网络中的每个边，任意两个不同的节点之间至多只能有一条边，每个节点不能与自身相连。当概率p=0那么重连永远不会发生，形成规则网络；如果概率p=1那么所有的连接都被重连了一次，最后得到完全随机网络。

观察集聚系数C=C(P)，平均路径长度L=L(P)与重连概率p的变化，发现在p从0变到1的过程中，L(P)下降得很快，而C(P)下降的比较慢。WS模型小世界网络同时具有特征路径长度短和集群程度高的特点，它们并不能从规则网络或随机网络中推导出来，因此引入随机重连概率p模拟了规则网络和随机网络之间的情况，验证了小世界网络中短路径的存在性。

生成WS模型主要有以下步骤：

1. 生成规则网络，N = 300，每个节点与周围最近的6个节点相连。
2. N=300;
3. degree = 6;
4. % break\_p = [0, power(10,(-4:0.1:0)), 1];
5. connection = zeros(N,N);
6. for i = 1:degree/2
7. connection = connection + diag(ones(1,N-i),i) + diag(ones(1,i),N-i);
8. end
9. 将规则网络的每个连接按照概率p从0.0001~1取值，并重新连接，保证两个不同的节点之间至多只能有一条边且每个节点不能与自身相连，形成新的网络。
10. reconnection = connection .\*rand(N,N);
11. for row = 1:N-1
12. for column = row+1:N
13. if((reconnection(row, column) > 0) && (reconnection(row, column) < p))
14. reconnection(row, column) = 0;
16. while(1)
17. new\_connection = randi([1 N],1);
19. if((new\_connection ~= row) && (reconnection(row, new\_connection) == 0) && (reconnection(new\_connection, row) == 0))
20. reconnection(new\_connection, row) = 1;
21. reconnection(row, new\_connection) = 1;
22. break;
23. end
24. end
25. end
26. end
27. end
28. reconnection = triu(reconnection);
29. reconnection = ceil(reconnection);
30. reconnection = reconnection + reconnection';
31. 统计平均路径长度与集聚系数随重连概率的变化。
32. %求一个网络的平均【最短距离路径】：节点对之间距离之和/节点对数目。
33. %其中，不可达两点距离为0，节点自身与自身距离为0，节点对数目为(N\*(N-1)/2)
34. D = reconnection; %D为距离矩阵
35. D(find(D==0))=inf;
36. for k=1:N %Floyd算法求解任意两点的最短距离
37. for e=1:N
38. for f=1:N
39. if D(e,f)>D(e,k)+D(k,f)
40. D(e,f)=D(e,k)+D(k,f);
41. end
42. end
43. end
44. D(k,k)=0;
45. end
46. D(find(D==inf))=0;
47. average\_shortest\_path\_length =(sum(D(:))/2)/(N\*(N-1)/2);
48. %求一个网络的平均集聚系数：所有节点的CC之和/节点数目
49. %一个节点的CC=邻居实际相连的边/邻居间应该相连的边=邻居实际相连的边/（di\*（di-1）/2）
50. %其中，di为节点i的度
51. %所以，算节点CC的方法二：以节点i的邻居们为节点，构造子图。子图中1的数目的一半就是CC
52. con\_matrix = reconnection;
53. sum\_CC = zeros(N,1);
54. a\_CC = zeros(N,1);
55. for k=1:N
56. num\_k=sum(con\_matrix(k,:));
58. if num\_k==0||num\_k==1 %若节点i度为0或1，则其没有CC
59. sum\_CC(k)=0;
60. else
61. for e=1:N
62. for f=e+1:N
63. if con\_matrix(k,e)+con\_matrix(k,f)==2 && con\_matrix(e,f)==1
64. sum\_CC(k)=sum\_CC(k)+1;
65. end
66. end
67. end
68. a\_CC(k)=sum\_CC(k)/((num\_k)\*(num\_k-1)/2);
69. end
70. end
71. ave\_CC = sum(a\_CC)/N;
72. c = ave\_CC;
73. d = average\_shortest\_path\_length;
74. 重复100次上述过程，获取平均路径长度与集聚系数的平均值。
75. p = power(10,(-4:0.1:0));
76. [c\_0 d\_0] = calculate(0);
77. buf = zeros(2, length(p));
78. for i = 1:length(p)
79. i
80. repeat = 50;
81. average\_buf = zeros(2,repeat);
82. for j = 1:repeat
83. [average\_buf(1,j) average\_buf(2,j)]= calculate(p(i));
84. end
85. buf(:,i)= mean(average\_buf,2);
86. end
87. buf\_normalized(1,:) = buf(1,:)/c\_0;
88. buf\_normalized(2,:) = buf(2,:)/d\_0;
89. title('平均最短路径和集聚系数随重连概率的变化');
90. semilogx(p, buf\_normalized','\*');
91. xlim([0 1.1]);
92. ylim([0 1.1]);
93. legend('C(p)/C(0)','D(p)/D(0)');
94. set(gca,'xticklabel',get(gca,'xtick'));
95. xlabel('重连概率 p');

