%%%实现个体的交叉变异matlab代码:

%从桥梁传感器103\*50模态数据中，共103个点中，选出zh个点进行组合,每一个个体的每一维是离散的，因此

%为103条记录进行编号

function ga\_main(zh)

%导入模态文件

format long

file='C:\Users\my\_pc\_2017\_1\_2\Desktop\node\_trans.txt';

mt=load(file);

m=mt;

[rows,cols]=size(mt);%获得模态矩阵的行和列,第一列表示为节点编的编号，方便找到对应的节点，第二列表示实际节

%定义目标函数，目标函数为r=obj\_val()

%节点组合个数

n=zh;

%遗传代数

ge=100;

%种群规模

gm=50;

%种群代沟

gg=0.7;

%种群变异概率

gb=0.0125;

%交叉概率

gj=0.7;

%重插入概率

gc=1;

%寻优结果的初始

%设置初始化随机数发生器

rand('state',sum(clock));

for k=1:gm %通过改良圈算法选取初始种群

    cl=randperm(103,103);%生成20个1-103的随机且不重复的数据，即生成初始解

  %保存种群所有的组合

  J(k,1:103)=cl;

end

 %disp(J);

 %计算初始种群的适应度值

 j1=J(:,1:n);

 objvalue=obj\_val(j1,m);

 J=J/103;%把整数序列转换成[0,1]区间上的实数，即转换成染色体编码

 for k=1:ge%该层循环进行遗传算法的操作

     A=J;%交配产生子代A的初始染色体

     if rand<gj

     c=randperm(gm);%产生下面交叉操作的染色体对

    % disp(c);

     for i=1:2:gm

         F=1+floor((n-1)\*rand(1));%产生交叉操作的的地址

         temp=A(c(i),[F:n]);%中间变量的保存值

         A(c(i),[F:n])=A(c(i+1),[F:n]);%交叉操作

         A(c(i+1),F:n)=temp;

     end

     end

      B=[];

      if rand<gb

     by=[];%为了防止下面产生空地址，这里先初始化

     while ~length(by)

         by=find(rand(1,gm)<0.1);%产生变异操作的地址

     end

     B=A(by,:);%产生变异操作的初始染色体

     for j=1:length(by)

         bw=sort(1+floor((n-1)\*rand(1,3)));%产生变异操作的3个地址

         B(j,:)=B(j,[1:bw(1)-1,bw(2)+1:bw(3),bw(1):bw(2),bw(3)+1:end]);%交换位置

     end

      end

     G=[J;A;B];%父代和子代种群和在一起

     %disp(G);

     [SG,ind1]=sort(G,2);%把染色体翻译成1...20的序列ind1,此处需要修改

     num=size(G,1);

     long=zeros(1,num);

     g1=ind1(:,1:n);

     %fprintf('交叉变异后的种群：\n');

     %disp(g1);

     objvalue=obj\_val(g1,m);

     long=objvalue;

     [slong,ind2]=sort(long);%对路径长度从小到大排序

     J=G(ind2((end+1)-gm:end),:);%精选前m个较大的组合对应的染色体

 end

 [SG1,ind3]=sort(J,2);

  fprintf('最终的索引：\n');

 disp(ind3);

 path=ind3(gm,:),

 path=path(:,1:n),

 objvalue=obj\_val(path,m);

 pp=zeros(1,n);

 for k1=1:n

  pp(k1)=m(path(k1),2);

 end

 fprintf('最终的桥梁节点组合：\n');

 pp=sort(pp);

 max=objvalue;

 disp(pp)

end

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%%计算适应度值

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%定义最佳组合节点的适应度求解函数

function [objvalue] =obj\_val(obj,m)%obj表示传进来的目标矩阵，m表示总共的节点数据，输出为适应度值

     [rows,cols]=size(obj);

     [rs,cs]=size(m);

     %将对应节点的模态值放进一个矩阵里

     val=zeros(rows);

     for i=1:rows

         sum2=0;%保存

         for j=1:cols %获得一个个体对应的目标模态矩阵

             temp(j,:)=m(obj(i,j),3:end);%

         end

       %获得存储个体对应的模态矩阵的行数和列数

       [rt,ct]=size(temp);

   %计算目标模态矩阵的累积和

   for k1=1:ct

       for k2=1:ct

          sum2=sum2+abs(temp(:,k1)'\*temp(:,k2));

       end

   end

      a(i)=sum2;

     end

      objvalue=a,

end

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

主函数部分

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%改变n的个数，指定生成的目标节点个数

clc;

clear;

n=10;%需要组合的节点的个数

for i=1:10

fprintf('节点个数=%d时第%d次试验结果为\n',n,i);

    ga\_main(n)%调用遗传算法，求得目标节点

end