## 实验内容：用遗传算法求解函数最优化问题

### 1. 实验时长：2学时

### 2. 实验目的：

1）熟悉遗传算法的原理、特征、结构及计算过程。

2）通过实验，掌握用遗传算法求解函数最优化问题的工作原理。

3) 了解遗传算法各参数对算法效果的影响。

4）掌握用Matlab实现遗传算法求解函数最优化问题的方法。

### 3. 实验内容：

遗传算法（Genetic Algorithm，GA）最早是由美国的 John holland于20世纪70年代提出,该算法是根据大自然中生物体进化规律而设计提出的。是模拟[达尔文](https://baike.baidu.com/item/%E8%BE%BE%E5%B0%94%E6%96%87/23890)[生物进化论](https://baike.baidu.com/item/%E7%94%9F%E7%89%A9%E8%BF%9B%E5%8C%96%E8%AE%BA/143686)的[自然选择](https://baike.baidu.com/item/%E8%87%AA%E7%84%B6%E9%80%89%E6%8B%A9/1800930)和[遗传学](https://baike.baidu.com/item/%E9%81%97%E4%BC%A0%E5%AD%A6/233918)机理的[生物进化](https://baike.baidu.com/item/%E7%94%9F%E7%89%A9%E8%BF%9B%E5%8C%96/3311428)过程的计算[模型](https://baike.baidu.com/item/%E6%A8%A1%E5%9E%8B/1741186)，是一种通过模拟自然进化过程搜索[最优解](https://baike.baidu.com/item/%E6%9C%80%E4%BC%98%E8%A7%A3/5208902)的方法。该算法通过数学的方式,利用[计算机仿真](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E4%BB%BF%E7%9C%9F/2056561)运算,将问题的求解过程转换成类似生物进化中的[染色体](https://baike.baidu.com/item/%E6%9F%93%E8%89%B2%E4%BD%93/195881)基因的交叉、变异等过程。在求解较为复杂的组合优化问题时,相对一些常规的优化算法,通常能够较快地获得较好的优化结果。遗传算法已被人们广泛地应用于组合优化、机器学习、信号处理、自适应控制和人工生命等领域。

1) 遗传算法的构成要素：

1. 染色体编码方法：

由于遗传算法不能直接处理问题空间的参数,因此必须通过编码将要求解的问题表示成遗传空间的染色体或者个体。这一转换操作就叫做编码，也可以称作（问题的）表示（representation）。  

评估编码策略常采用以下3个[规范](https://baike.baidu.com/item/%E8%A7%84%E8%8C%83" \t "_blank)：

a)[完备性](https://baike.baidu.com/item/%E5%AE%8C%E5%A4%87%E6%80%A7)(completeness):问题空间中的所有点(候选解)都能作为GA空间中的点(染色体)表现。

b)健全性(soundness): GA空间中的染色体能对应所有问题空间中的候选解。

c)非冗余性(nonredundancy):染色体和候选解一一对应。

1. 个体适应度评价：

适应度函数，是表示某一个体对[环境](https://baike.baidu.com/item/%E7%8E%AF%E5%A2%83)的适应能力，也表示该个体繁殖后代的能力。遗传算法的适应度[函数](https://baike.baidu.com/item/%E5%87%BD%E6%95%B0)也叫[评价函数](https://baike.baidu.com/item/%E8%AF%84%E4%BB%B7%E5%87%BD%E6%95%B0)，是用来判断群体中的个体的优劣程度的指标，它是根据所求问题的[目标函数](https://baike.baidu.com/item/%E7%9B%AE%E6%A0%87%E5%87%BD%E6%95%B0)来进行评估的。

1. 遗传算子： 选择、交叉、变异

[选择运算](https://baike.baidu.com/item/%E9%80%89%E6%8B%A9%E8%BF%90%E7%AE%97)：将选择算子作用于群体。选择的目的是把优化的个体直接遗传到下一代或通过配对交叉产生新的个体再遗传到下一代。选择操作是建立在群体中个体的[适应度](https://baike.baidu.com/item/%E9%80%82%E5%BA%94%E5%BA%A6)评估基础上的。

交叉运算：将交叉算子作用于群体。遗传算法中起核心作用的就是交叉算子。

[变异运算](https://baike.baidu.com/item/%E5%8F%98%E5%BC%82%E8%BF%90%E7%AE%97)：将变异算子作用于群体。即是对群体中的个体串的某些[基因座](https://baike.baidu.com/item/%E5%9F%BA%E5%9B%A0%E5%BA%A7)上的基因值作变动。

群体P(t)经过选择、交叉、[变异运算](https://baike.baidu.com/item/%E5%8F%98%E5%BC%82%E8%BF%90%E7%AE%97" \t "_blank)之后得到下一代群体P(t+1)。

1. 基本遗传算法的运行参数：

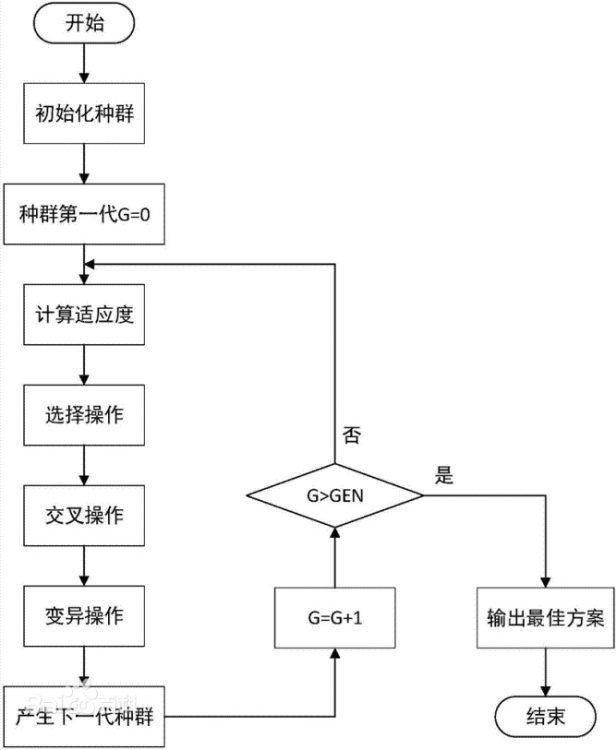
M： 种群大小，即群体中所含个体的数量

G: 遗传算法的终止进化代数

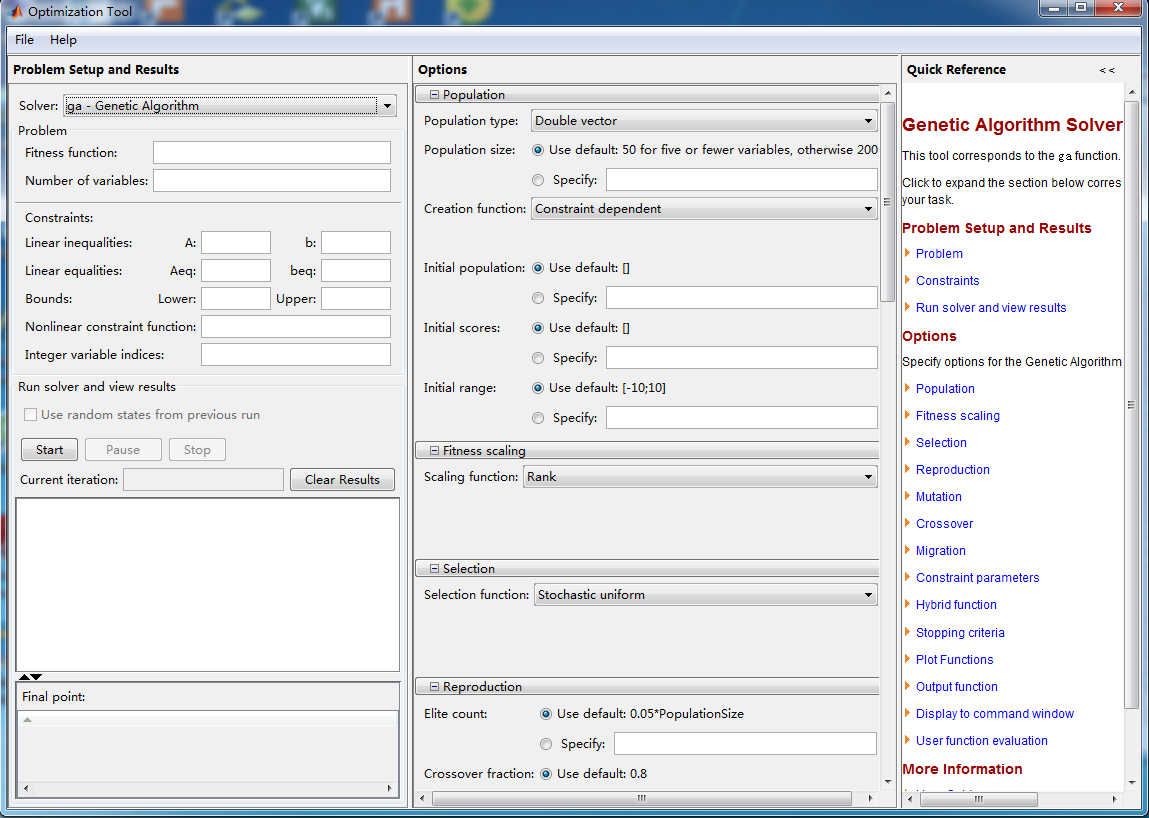
Pc： 交叉概率

Pm：变异概率

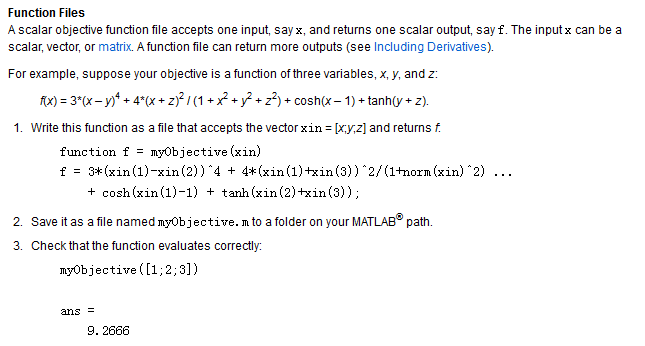
2) 遗传算法的应用步骤



3）命令行输入：optimtool



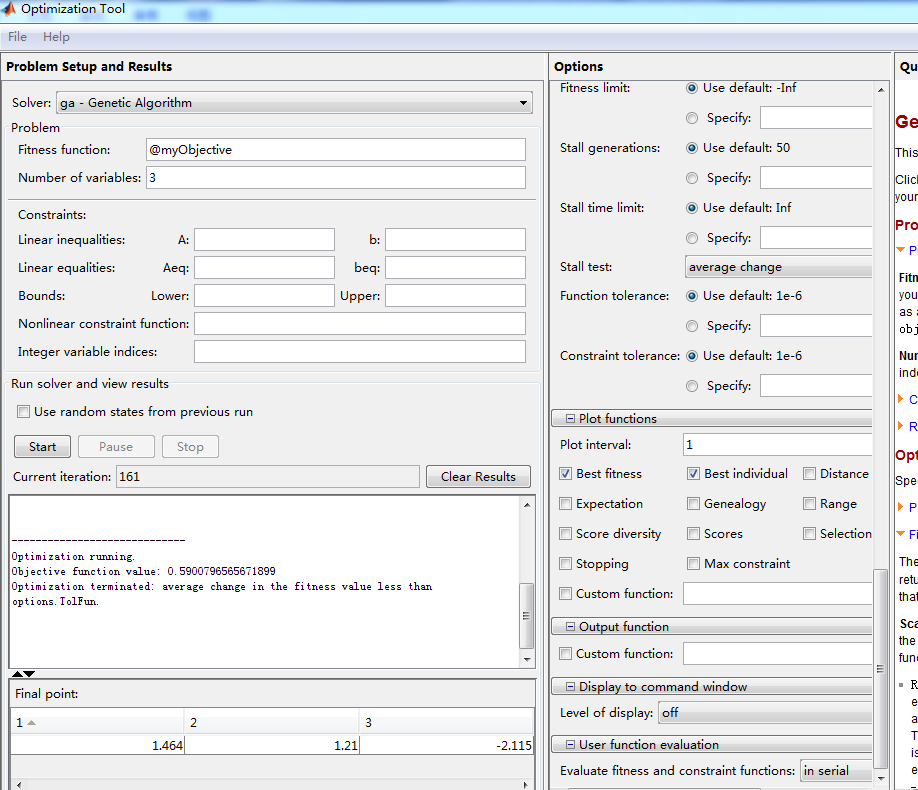
适应度函数：



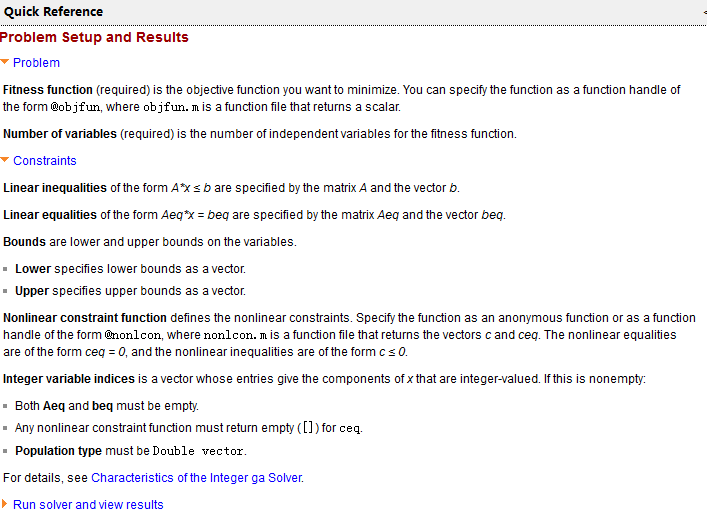
function f = myObjective(xin)

f = 3\*(xin(1)-xin(2))^4 + 4\*(xin(1)+xin(3))^2/(1+norm(xin)^2) ...

+ cosh(xin(1)-1) + tanh(xin(2)+xin(3));



有关其他参数，约束等定义如下图所示：



### 4. 遗传算法的代码实现、

### MATLAB谢菲尔德遗传算法工具箱使用

### 种群表示和初始化

函数 bs2rv:  二进制串到实值的转换

             Phen=bs2rv(Chrom,FieldD)

             FieldD=[len, lb, ub, code, scale, lbin, ubin]

len是字串的长度，lb和ub是每个变量的下界和上界，lbin和ubin指明表示范围中是否包含边界，0不包括，1包括。

             code(i)=1为标准的二进制编码，code(i)=0为格雷编码

             scale(i)=0为算术刻度，scale(i)=1为对数刻度

函数 crtbp:  创建初始种群

            [Chrom,Lind,BaseV]=crtbp(Nind,Lind)

            [Chrom,Lind,BaseV]=crtbp(Nind,BaseV)

            [Chrom,Lind,BaseV]=crtbp(Nind,Lind,BaseV)

             Nind指定种群中个体的数量，Lind指定个体的长度，Basev与base是一样的，返回的是个体的各个位的进制数

函数 crtrp:  创建实值原始种群

            Chrom=crtrp(Nind,FieldDR)

### 适应度计算

函数 ranking:  基于排序的适应度分配（此函数是从最小化方向对个体进行排序的）

              FitV=ranking(ObjV)

              FitV=ranking(ObjV, RFun)

              FitV=ranking(ObjV, RFun, SUBPOP)

              Rfun(1)线性排序标量在[1 2]间，代表的是选择压差，比如压差为2，那就限制了Objv的结果fitv在0-2之间，如果是1.5，就在0.5-1.5之间，结果的最大值与最小值加起来就是2，非线性排序在[1 length(ObjV)-2]

              Rfun(2)指定排序方法，0为线性排序，1为非线性排序

              SUBPOP指明ObjV中子种群的数量，默认为1

   ranking(A)= 2.0000    1.4000    0.6000         0    1.4000    0.6000，所以如果是求最大值，就得-A，求最小值就直接A就可以了，因为目标值越大分配的适应度越小，目标值越小，分配的适应度越大。ranking的结果FitV只在select选择的时候有用到，这只是一种个体选择概率的常用分配方法之一

关于个体选择概率的常用分配方法参考：

### 选择

高级函数 select:  从种群中选择个体

                SelCh=select(SEL\_F, Chrom, FitnV)

                SelCh=select(SEL\_F, Chrom, FitnV, GGAP)

                SelCh=select(SEL\_F, Chrom, FitnV, GGAP, SUBPOP)

                SEL\_F是一字符串，为一低级选择函数名，如rws或sus

                GGAP指出了代沟，默认为1；也可大于1，允许子代数多于父代的数量

SUBPOP是指子代种群的个数

rws: 轮盘赌选择

     NewChrIx=rws(FitnV, Nsel) 使用轮盘赌选择从一个种群中选择Nsel个个体

     NewChrIx 是为育种选择的个体的索引值

sus:  随机遍历抽样

     NewChrIx=sus(FitnV, Nsel)

关于选择方法的介绍及比较参考：http://wenku.baidu.com/link?url=fyc9LUjso9IJmYhEKd7fnLl\_25CsTdGZEm1CyCtHaDj2DtYe5K9IQCcY7DhcHD-dTlsgdgjobsImdOQUr4MmKm9CytLBsUU1oLCBLyn4uVO

### 交叉

高级函数 recombin:  重组个体

                    NewChrom=recombin(REC\_F, Chrom)

                    NewChrom=recombin(REC\_F, Chrom, RecOpt)

                    NewChrom=recombin(REC\_F, Chrom, RecOpt, SUBPOP)

chrom是待交叉的种群，recopt是交叉概率，subpop是一个决定子种群个数的可选参数，默认是1，chrom中所有子种群的大小必须一致。

                    REC\_F是包含低级重组函数名的字符串，交叉方式，例如recdis,recint,reclin,xovdp, xovdprs, xovmp, xovsh, xovshrs, xovsp, xovsprs

recdis:    离散重组

          NewChrom=recdis(OldChorm)

recint:    中间重组

          NewChrom=recint(OldChorm)

reclin:    线性重组

          NewChrom=reclin(OldChorm)

xovdp:    两点交叉

          NewChrom=xovdp(OldChrom, XOVR)

          XOVR为交叉概率， 默认为0.7

xovdprs:  减少代理的两点交叉

          NewChrom=xovdprs(OldChrom, XOVR)

xovmp:   多点交叉

          NewChrom=xovmp(OldChrom, XOVR, Npt, Rs)

          Npt指明交叉点数， 0 洗牌交叉；1 单点交叉；2 两点交叉；  默认为0

          Rs指明使用减少代理， 0 不减少代理；1 减少代理；  默认为0

xovsh:    洗牌交叉

          NewChrom=xovsh(OldChrom, XOVR)

xovshrs:  减少代理的洗牌交叉

          NewChrom=xovshrs(OldChrom, XOVR)

xovsp:    单点交叉

          NewChrom=xovsp(OldChrom, XOVR)

xovsprs:  减少代理的单点交叉

          NewChrom=xovsprs(OldChrom, XOVR)

### 变异

高级函数 mutate:  个体的变异

                  NewChorm=mutate(MUT\_F, OldChorm, FieldDR)

NewChorm=mutate(MUT\_F, OldChorm, FieldDR, MutOpt)

NewChorm=mutate(MUT\_F, OldChorm, FieldDR, MutOpt, SUBPOP)

MUT\_F为包含低级变异函数的字符串，例如mut, mutbga, recmut

**mut:  离散变异算子**

**NewChrom=mut(OldChorm, Pm)**

**NewChrom=mut(OldChorm, Pm, BaseV)**

**Pm为变异概率，默认为Pm=0.7/Lind，basev是染色体的各个位的进制数，目前只用过这个**

mutbga:  实值种群的变异（遗传算法育种器的变异算子）

NewChrom=mutbga(OldChorm, FieldDR)

          NewChrom=mubga(OldChorm, FieidDR, MutOpt)

          MutOpt(1)是在[0 1]间的重组概率的标量，默认为1

          MutOpt(2)是在[0 1]间的压缩重组范围的标量，默认为1（不压缩）

recmut:   具有突变特征的线性重组

          NewChrom=recmut(OldChorm, FieldDR)

          NewChrom=recmut(OldChorm, FieidDR, MutOpt)

### 重插入

函数 reins:  重插入子群到种群

            Chorm=reins(Chorm, SelCh)

            Chorm=reins(Chorm, SelCh, SUBPOP)

            Chorm=reins(Chorm, SelCh, SUBPOP, InsOpt, ObjVch)

            [Chorm, ObjVch]=reins(Chorm, SelCh, SUBPOP, InsOpt, ObjVch, ObjVSel)

Chorm, 父代种群，SelCh子代种群

            InsOpt(1)指明用子代代替父代的选择方法，0为均匀选择，1为基于适应度的选择，默认为0

            InsOpt(2)指明在[0 1]间每个子种群中重插入的子代个体在整个子种群的中个体的比率，默认为1

            ObjVch包含Chorm中个体的目标值，对基于适应度的重插入是必需的

            ObjVSel包含Selch中个体的目标值，如子代数量大于重插入种群的子代数量是必需的

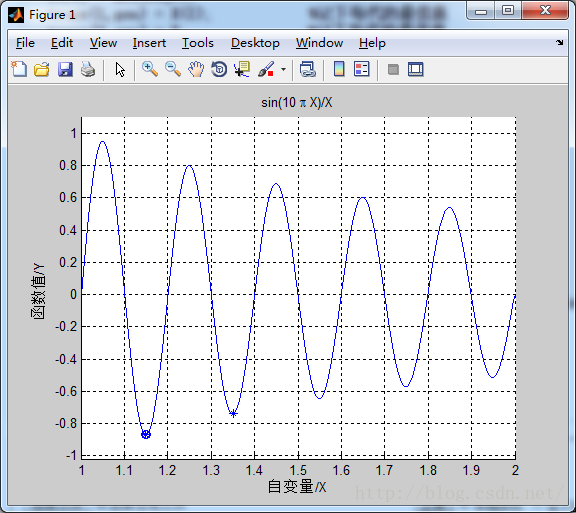
### 其他函数

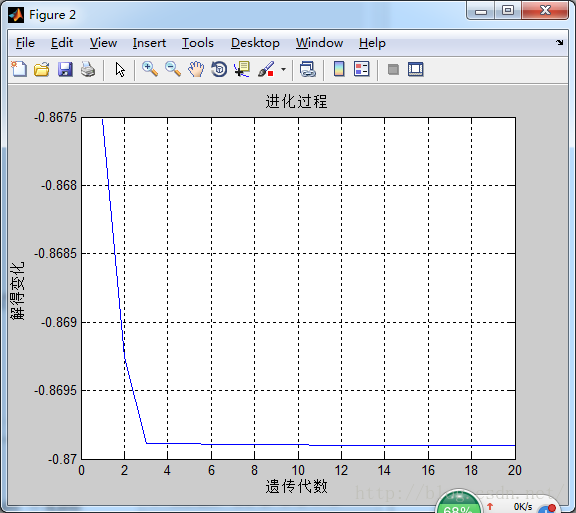
矩阵复试函数 rep:  MatOut=rep(MatIn, REPN)   REPN为复制次数

使用案例：

clc  
clear all  
close all  
%% 画出函数图  
figure(1);  
hold on;  
lb=1;ub=2; %函数自变量范围【1,2】  
ezplot('sin(10\*pi\*X)/X',[lb,ub]);   %画出函数曲线  
xlabel('自变量/X')  
ylabel('函数值/Y')  
%% 定义遗传算法参数  
NIND=40;        %个体数目  
MAXGEN=20;      %最大遗传代数  
PRECI=20;       %变量的二进制位数  
GGAP=0.95;      %代沟  
px=0.7;         %交叉概率  
pm=0.01;        %变异概率  
trace=zeros(2,MAXGEN);                        %寻优结果的初始值  
FieldD=[PRECI;lb;ub;1;0;1;1];                      %区域描述器  
Chrom=crtbp(NIND,PRECI);                      %初始种群  
%% 优化  
gen=0;                                  %代计数器  
X=bs2rv(Chrom,FieldD);                 %计算初始种群的十进制转换  
ObjV=sin(10\*pi\*X)./X;        %计算目标函数值  
while gen<MAXGEN  
   FitnV=ranking(ObjV);                               %分配适应度值  
   SelCh=select('sus',Chrom,FitnV,GGAP);              %选择，随机遍历选择  
   SelCh=recombin('xovsp',SelCh,px);                  %重组，单点交叉  
   SelCh=mut(SelCh,pm);                               %变异  
   X=bs2rv(SelCh,FieldD);               %子代个体的十进制转换  
   ObjVSel=sin(10\*pi\*X)./X;             %计算子代的目标函数值  
   [Chrom,ObjV]=reins(Chrom,SelCh,1,1,ObjV,ObjVSel); %重插入子代到父代，得到新种群  
   %chrom是父代种群，selch是子代，1，子种群数目都是1,1，基于适应度的选择，用子代代替父代适应度最小的个体  
   X=bs2rv(Chrom,FieldD);  
   gen=gen+1;                                             %代计数器增加  
   %获取每代的最优解及其序号，Y为最优解,I为个体的序号  
   [Y,I]=min(ObjV);  
   trace(1,gen)=X(I);                            %记下每代的最优值  
   trace(2,gen)=Y;                               %记下每代的最优值  
end  
figure(2);  
plot(trace(1,:),trace(2,:),'bo');                            %画出每代的最优点  
grid on;  
plot(X,ObjV,'b\*');   %画出最后一代的种群  
title('每代的最优点')  
%此时x中存的是最后一代种群的所有个体的十进制值  
hold off  
%% 画进化图  
figure(3);  
plot(1:MAXGEN,trace(2,:));  
grid on  
xlabel('遗传代数')  
ylabel('解的变化')  
title('进化过程')  
bestY=trace(2,end);  
bestX=trace(1,end);  
fprintf(['最优解:\nX=',num2str(bestX),'\nY=',num2str(bestY),'\n'])

运行结果：







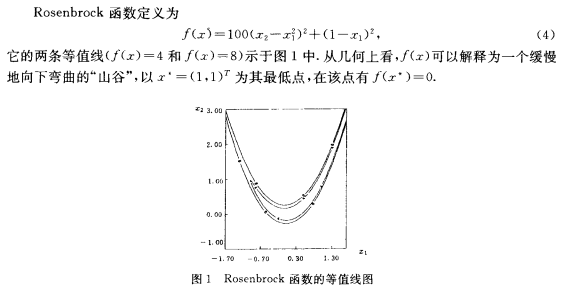
案例2：

clc  
clear all  
close all  
%% 画出函数图  
figure(1);  
lbx=-2;ubx=2; %函数自变量x范围【-2,2】  
lby=-2;uby=2; %函数自变量y范围【-2,2】  
ezmesh('y\*sin(2\*pi\*x)+x\*cos(2\*pi\*y)',[lbx,ubx,lby,uby],50);   %画出函数曲线  
hold on;  
%% 定义遗传算法参数  
NIND=40;        %个体数目  
MAXGEN=50;      %最大遗传代数  
PRECI=20;       %变量的二进制位数  
GGAP=0.95;      %代沟  
px=0.7;         %交叉概率  
pm=0.01;        %变异概率  
trace=zeros(3,MAXGEN);                        %寻优结果的初始值  
FieldD=[PRECI PRECI;lbx lby;ubx uby;1 1;0 0;1 1;1 1];                      %区域描述器  
Chrom=crtbp(NIND,PRECI\*2);                      %初始种群  
%% 优化  
gen=0;                                  %代计数器  
XY=bs2rv(Chrom,FieldD);                 %计算初始种群的十进制转换  
X=XY(:,1);Y=XY(:,2);  
ObjV=Y.\*sin(2\*pi\*X)+X.\*cos(2\*pi\*Y);        %计算目标函数值  
while gen<MAXGEN  
   FitnV=ranking(-ObjV);                              %分配适应度值，因为这里是求最大值，所以改成了负的  
   SelCh=select('sus',Chrom,FitnV,GGAP);              %选择  
   SelCh=recombin('xovsp',SelCh,px);                  %重组，单点交叉  
   SelCh=mut(SelCh,pm);                               %变异  
   XY=bs2rv(SelCh,FieldD);               %子代个体的十进制转换  
   X=XY(:,1);Y=XY(:,2);  
   ObjVSel=Y.\*sin(2\*pi\*X)+X.\*cos(2\*pi\*Y);             %计算子代的目标函数值  
   [Chrom,ObjV]=reins(Chrom,SelCh,1,1,-ObjV,-ObjVSel); %重插入子代到父代，得到新种群，因为此处是求最大值，所以目标值都是负的

  % 1，子种群的个数是1,1，基于适应度的选择，子代代替父代中适应度最小的个体  
   XY=bs2rv(Chrom,FieldD);  
   gen=gen+1;                                             %代计数器增加  
   %获取每代的最优解及其序号，Y为最优解,I为个体的序号  
   [Y,I]=max(ObjV);  
   trace(1:2,gen)=XY(I,:);                       %记下每代的最优值  
   trace(3,gen)=Y;                               %记下每代的最优值  
end  
plot3(trace(1,:),trace(2,:),trace(3,:),'bo');                            %画出每代的最优点  
grid on;  
plot3(XY(:,1),XY(:,2),ObjV,'bo');  %画出最后一代的种群  
hold off  
%% 画进化图  
figure(2);  
plot(1:MAXGEN,trace(3,:));  
grid on  
xlabel('遗传代数')  
ylabel('解的变化')  
title('进化过程')  
bestZ=trace(3,end);  
bestX=trace(1,end);  
bestY=trace(2,end);  
fprintf(['最优解:\nX=',num2str(bestX),'\nY=',num2str(bestY),'\nZ=',num2str(bestZ),'\n'])

### 5. 课堂作业题目：

1） 课堂作业： 采用GA代码版本，优化下列函数



### 6. 作业提交要求

1. 给出遗传算法求解函数最优化问题的设计过程和程序清单

2. 记录实验数据和曲线

3. 分析实验结果

备注： 作业提交Canvas系统，压缩文件请用 学号+姓名+GA01 来命名