**实验一 遗传算法解决函数优化问题**

**一、实验目的**

1. 掌握遗传算法的基本原理和步骤。
2. 熟练使用MATLAB软件编写遗传算法程序。
3. 利用遗传算法解决函数优化问题。

**二、实验原理**

遗传算法是一类随机优化算法，通过对染色体的评价和对染色体中基因的遗传操作，有效地指导搜索有希望改善优化质量的状态。标准遗传算法主要步骤可描述如下：

1、随机产生一组初始个体构成初始种群。

2、计算每一个体的适配值（fitness value，也称为适应度）。适应度值是对染色体(个体)进行评价的一种指标，是遗传算法进行优化所用的主要信息，它与个体的目标值存在一种对应关系。

3、判断算法收敛准则是否满足，若满足，则输出搜索结果；否则执行以下步骤。

4、根据适应度值大小以一定方式（例如赌轮法）执行选择操作。

5、按交叉概率*p*c执行交叉操作。

6、按变异概率*p*m执行变异操作。

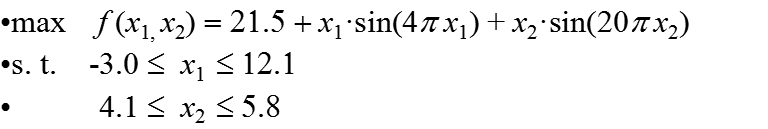
7、返回步骤②。

**三、实验内容**

1. 选择测试函数，函数可以自定义或者选择以下函数：

1）一元函数求最大值的优化问题 

2）二元函数求最大值的优化问题



2. 根据精度要求确定二进制位串编码码长。

3. 确定初始种群规模N，随机生成初始种群。

1. 编写解码程序，并计算适应度函数。
2. 确定算法终止条件（指定迭代次数或者其他条件）
3. 由赌轮法选择进入下一代的个体。
4. 随机选择两个个体，选择单点交叉位置，按照交叉概率*p*c执行交叉操作，得到两个新的个体。
5. 随机选择一个个体，按照变异概率*p*m执行变异操作，得到一个新的个体。
6. 记录种群中的个体及截止到目前最好的适应度。
7. 达到迭代要求，输出函数最优值和最优解。

**四、预习要求**

1. 认真阅读教材中遗传算法的基本原理与步骤。
2. 根据选择的待优化函数，确定编码方案，编写解码程序和适应度计算函数文件。
3. 编写赌轮法选择程序。

**五、实验要求**

1. 控制参数（精度要求、种群规模、迭代次数、交叉和变异概率）自选，绘制流程图。

2. 上机编写程序并调试运行程序。

3. 对比单点交叉、两点交叉和均匀交叉对算法性能的影响（选做）。

4. 对比赌轮法和随机采样法选择对算法性能的影响（选做）。

5. 绘制种群截止目前最好的适应度和每代平均适应度的变化曲线。

6. 根据实验结果，撰写实验报告。

**六、实验报告**

1. 实验报告包括实验目的、实验原理、实验内容、实验程序、实验结果与实验分析、实验心得体会。

2. 给出完整的源程序，并对源程序中的关键内容或语句给出必要的文字注释。

3. 给出实验结果（实验结果有图的话可截图打印）。

4. 利用实验数据，分析并解答以下问题，

* 1. 遗传算法中变异概率的设置对求解结果的影响。
  2. 遗传算法中交叉概率的设置对求解结果的影响。
  3. 遗传算法中种群规模的设置对求解结果的影响。

5. 分析遗传算法的优缺点，并提出改进意见。

6. 本实验的心得体会。

附1：

N

N

计算各个体的适配值（适应度）

算法收敛准则满足？

Y

random[0,1]<*P****c***?

选择

输出搜索结果

交叉

Y

random[0,1]<*P****m***?

变异

N

随机产生初始种群

Y

图1.1 标准遗传算法流程图

附2：例程

clear all

N=10;pc=0.95;pm=0.1;

per=4;

a=-2;b=1;

L=ceil(log2((b-a)\*10^per));

s=round(rand(N,L));

maxf=0;

for i=1:N

x=a+bin2dec(num2str(s(i,:)))\*(b-a)/(2^L-1);

y(i)=f(x);

if y(i)>maxf

maxf=y(i);

opmx=x;

end

end

T=100;

for t=1:T

%select

p=y/sum(y);

q=cumsum(p);

for i=1:N

temp=rand(1);

Ind=find(q>=temp);

s2(i,:)=s(Ind(1),:);

end

%crossover

for i=1:N

if rand(1)<pc

I=1+fix((N-1)\*rand(1));

J=1+fix((N-1)\*rand(1));

if I~=J

k=round((L-1)\*rand(1));

temp1=s2(I,k+1:end);

temp2=s2(J,k+1:end);

s2(I,k+1:end)= temp2;

s2(J,k+1:end)= temp1;

end

end

end

%mututation

for i=1:N

if rand(1)<pm

I=1+fix((N-1)\*rand(1));

k=1+fix((L-1)\*rand(1));

s2(I,k)=1-s2(I,k);

end

end

s=s2;

for i=1:N

x=a+bin2dec(num2str(s(i,:)))\*(b-a)/(2^L-1);

y(i)=f(x);

if y(i)>maxf

maxf=y(i);

opmx=x;

end

end

max\_f(t)=maxf;

mean\_f(t)=mean(y);

end

plot(1:T,max\_f,'b',1:T,mean\_f,'g')

maxf

opmx