

matlab代码如下：

%%%%%%%%先画图看看函数形状%%%%%%%%%%%%%%%%%

clear all; %清除所有变量

close all; %清图

clc; %清屏

x=-2:0.01:2;

y=func1(x);

plot(x,y)

xlabel('x')

ylabel('f(x)')

title('f(x)=sinc(x)')

%%%%%%%%标准遗传算法求函数极值%%%%%%%%%%

%%%%%%%%%%%初始化参数%%%%%%%%%%%%%%%%%%

clear all; %清除所有变量

clc; %清屏

NP=30; %种群数量

L=16; %二进制位串长度

Pc=0.3; %交叉率

Pm=0.01; %变异率

G=400; %最大遗传代数

Xs=2; %上限

Xx=-2; %下限

f=randi([0,1],NP,L);%随机获得初始种群（二维数组）30\*16

%%%%%%%%遗传算法循环%%%%%%%%%%%%%%%%%%

for k=1:G

%%%%%%%将二进制解码为定义域范围内十进制%%%%%%%

for i=1:NP %对种群中每个个体

U=f(i,:); %求种群中每个个体的染色体二进制编码数组

m=0;

for j=1:L %遍历某个个体的每一位

m=U(j)\*2^(j-1)+m;%求出该个体的二进制编码的十进制数

end

x(i)=Xx+m\*(Xs-Xx)/(2^L-1);%映射到所要求的的自变量区间上

Fit(i)=func1(x(i));%调用适应度函数

end

maxFit=max(Fit); %定义适应度中的最大值

minFit=min(Fit); %定义适应度中的最小值

rr=find(Fit==maxFit);

fBest=f(rr(1,1),:); %得到第k代最优个体的染色体编码数组

xBest=x(rr(1,1)); %得到最优个体对应十进制映射数值

Fit=(Fit-minFit)/(maxFit-minFit);%归一化适应度函数值[0 1]

%%%%%%%基于轮盘赌的复制操作%%%%%%%%%%

sum\_Fit=sum(Fit); %定义适应度函数值总和

fitvalue=Fit./sum\_Fit;

fitvalue=cumsum(fitvalue)

ms=sort(rand(NP,1))

fiti=1; %计数变量，表示原种群中当前被比较的个体序号

newi=1; %计数变量，表示现在被选择进入下一代的个体序号

while newi<=NP&&fiti<=NP

if (ms(newi))<fitvalue(fiti)

nf(newi,:)=f(fiti,:);

newi=newi+1; %进入下一代个体数加1

else

fiti=fiti+1; %若不满足，则该个体不被选择，进入下一个个体的判断

end

end

%%%%%%%%基于概率的交叉操作%%%%%%%%

for i=1:2:NP %步长为2，代表从原种群数组中选取相邻两个为一对考虑是否进行交叉操作

p=rand; %生成【0,1】随机小数

if p<Pc %若p小于交叉概率

q=randi([0,1],1,L); %生成一条（0,1）分布的二进制数串

for j=1:L %对该数串上的每一位

if q(j)==1 temp=nf(i+1,j);

nf(i+1,j)=nf(i,j);

nf(i,j)=temp; %上三步完成了第j位交叉互换

end

end

end

end

%%%%%%%%基于概率的变异操作%%%%%%%%%

i=1;

while i<=round(NP\*Pm) %round表示四舍五入法取整，表示总共要对NP\*Pm个个体进行变异

h=randi([1,NP],1,1); %随机选取一条需要变异的染色体

for j=1:round(L\*Pm) %在需要变异的某条染色体总共进行L\*Pm个基因变异

g=randi([1,L],1,1); %随机选取需要变异的基因序号

nf(h,g)=~nf(h,g); %取反完成变异

end

i=i+1;

end

f=nf;

f(1,:)=fBest; %保留最优个体在新种群中

trace(k)=maxFit; %把第k代最优适应度保存到数组trace中

end

disp(['最终函数最大值点为',num2str(xBest)])

disp(['最终函数最大值为',num2str(func1(xBest))])

figure

plot(trace)

xlabel('迭代次数')

ylabel('目标函数值')

axis([0,10,0,2]);

title('适应度进化曲线')

%%%%%%%%适应度函数%%%%%%%%%%

function result=func1(x) %定义函数名为func1和返回形式参数名为result的适应度函数

fit=0.4+sinc(4\*x)+1.1\*sinc(4\*x+2)+0.8\*sinc(x-2)+0.7\*sinc(6\*x-4); %直接把目标函数作为适应度函数

result=fit;

end

%%%%%%sinc函数%%%%%%%%%%%%

function y=sinc(x)

y=sin(pi\*x)./(pi\*x);

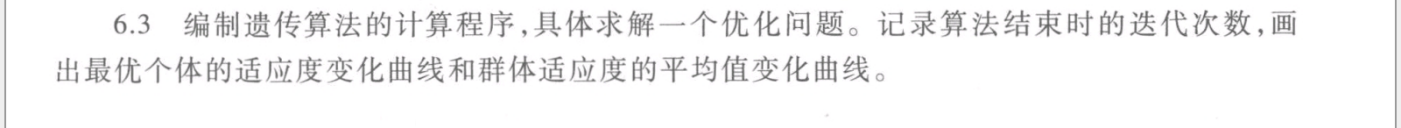
end

原函数图像：

遗传matlab代码运算结果：

最终函数最大值点为-0.50852

最终函数最大值为1.611



求解函数y=sin(2x)+cos(2x)+x在[0，20]的最大值;

群体规模50，

Pc=0.8，

Pm=0.1，

迭代次数为100

代码如下：

clc,clear

N = 50;

L = 20;

pc =0.8;

pm =0.1;

g =100;

xs =20;

xx =0;

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

for k =1:g

for i = 1:N

u =f (i,:);

m = 0;

for j =1:L

m = u(j)\*2^(j-1)+m;

end

x(i) = xx + m\*(xs-xx)/(2^L-1);

fit(i) = func1(x(i));

end

maxfit = max(fit);

minfit = min(fit);

rr = find(fit==maxfit);

fbest = f(rr(1,1),:);

xbest = x(rr(1,1));

fit = (fit-minfit)/(maxfit-minfit);

%轮盘赌赋值操作

sum\_fit = sum(fit);

fitvaule = fit./sum\_fit;

fitvaule = cumsum(fitvaule);

ms = sort(rand(N,1));

fiti = 1;

newi = 1;

while newi <= N

if (ms(newi)) < fitvaule(fiti)

nf(newi,:) = f(fiti,:);

newi = newi +1;

else

fiti = fiti +1;

end

end

%概率交差操作

for i = 1:2:N %选择两个优秀个体

p = rand;

if p <pc

q = round(rand(1,L));

for j =1:L

if q(j) == 1

temp = nf(i+1,j);

nf(i+1,j) = nf(i,j);

nf(i,j) = temp;

end

end

end

end

%变异操作

i =1;

while i <= round(N\*pm)

h = randi(N);

for j = 1:round(L\*pm)

g = randi(L);

nf(h,g) = ~nf(h,g);

end

i = i+1;

end

f = nf;

f(1,:) =fbest;

value(k) = maxfit;

avg(k)=

end

disp(['最终函数最大值点为',num2str(xbest)])

disp(['最终函数最大值为',num2str(func1(xbest))])

xbest;

xx = 0:0.1:20;

y = sin(2\*xx)+cos(3\*xx)+xx;

figure

plot(xx,y)

figure

plot(value)

function result = func1(x)

fit = sin(2\*x)+cos(3\*x)+x;

result = fit;

end

1原函数图像:

代码运行结果:

最终函数最大值点为19.1895

最终函数最大值为20.3417

2最有个体的适应度变化曲线：

群体适应度的平均值变化曲线：

3

给定的迭代次数为100，根据图像10代之后群体的平均适应度达到了稳定。