**实验二 进化策略和进化规划解决函数优化问题**

**一、实验目的**

1. 掌握进化策略和进化规划的基本原理和步骤。
2. 熟练使用MATLAB软件编写进化策略算法程序。
3. 利用进化策略和进化规划解决函数优化问题。

**二、实验原理**

进化策略和进化规划是一种模仿自然进化原理求解优化问题的算法，对求解实值优化问题有一定的优势。通过对目标变量和策略参数同时进行重组和变异，快速地搜索最优解。

进化策略算法主要步骤可描述如下：

1、初始化种群和策略参数。

2、计算每一个体的适配值（fitness value，也称为适应度）。

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、执行重组操作。

5、执行变异操作。

6、计算每一个体的适配值（fitness value，也称为适应度）。

7、选择下一代个体。

8、返回步骤③。

进化规划算法主要步骤可描述如下：

1、初始化种群和策略参数。

2、计算每一个体的适配值（fitness value，也称为适应度）。

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、执行变异操作。

5、计算每一个体的适配值（fitness value，也称为适应度）。

6、选择下一代个体。

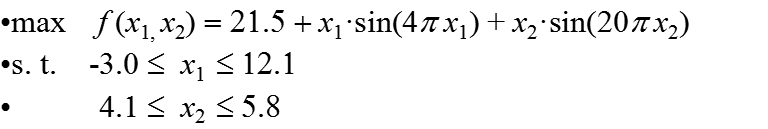
7、返回步骤③。

**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. 确定初始种群规模N和编码方案，生成初始种群和初始策略参数。

3. 确定算法终止条件（指定迭代次数或者其他条件）

1. 随机选择父代个体，对于进化策略执行交叉重组（离散或中值）和变异操作，对于胡不归规划执行变异操作。生成满足约束条件的若干个子代。
2. 进行种群适应度评价，由进化策略的及进化规划的随机型P竞争选择策略选择进入下一代的个体。

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. 认真阅读教材中进化策略和进化规划的基本原理与步骤。
2. 根据选择的待优化函数，编写适应度计算函数文件。
3. 编写重组和变异程序。

**五、实验要求**

1. 控制参数（种群规模、迭代次数、重组算子）自选，绘制流程图。

2. 上机编写程序并调试程序。

3. 对比加号策略和逗号策略对进化策略算法性能的影响。

4. 对比p值对进化规划算法性能影响。

5. 绘制种群每代最好的适应度和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. 进化策略中种群规模设置、选择策略对求解结果的影响。
  2. 进化策略中的取值对求解结果的影响。
  3. 进化规划中P的取值对求解结果的影响。

5. 分析进化策略和进化规划的优缺点，并提出改进意见。

6. 本实验的心得体会。

附1：进化策略例程（逗号策略）（μ,λ）改成(μ+λ)程序第33行

clear

clc

N=10;

x1=15.1\*rand(1,N)-3;

x2=1.7\*rand(1,N)+4.1;

X=[x1;x2];

sigma=rand(2,N);

T=50;

maxf=0;

for t=1:T

lamda=1;

while lamda<=7\*N

pos=1+fix(rand(1,2)\*(N-1));

pa1=X(:,pos(1));

pa2=X(:,pos(2));

if rand()<0.5

o(1)=pa1(1);

else

o(1)=pa2(1);

end

if rand()<0.5

o(2)=pa1(2);

else

o(2)=pa2(2);

end

sigma1=0.5\*(sigma(:,pos(1))+sigma(:,pos(2)));

Y=o'+sigma1.\*randn(2,1);

if Y(1)>=-3 & Y(1)<=12.1 & Y(2)>=4.1 & Y(2)<=5.8

offspring(:,lamda)=Y;

lamda=lamda+1;

end

end

U=[offspring];

for i=1:size(U,2)

eva(i)=f(U(:,i));

end

[m\_eval,I]=sort(eva);

I1=I(end-N+1:end);

X=U(:,I1);

if m\_eval(end)>maxf

maxf=m\_eval(end);

opmx=U(:,end);

end

max\_f(t)=maxf;

mean\_f(t)=mean(eva(I1));

end

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

opmx

maxf

附2：进化规划例程

clear

clc

N=20;

x1=15.1\*rand(1,N)-3;

x2=1.7\*rand(1,N)+4.1;

X=[x1;x2];

sigma=[3;3];

gen=1;

maxf=0;

while gen<100

i=0;

while i<N

if sigma(1)<0;

sigma(1)=0.001;

else

sigma(1)=sigma(1)+sqrt(sigma(1))\*randn(1);

end

if sigma(2)<0;

sigma(2)=0.001;

else

sigma(2)=sigma(2)+sqrt(sigma(2))\*randn(1);

end

X0=X(:,i+1)+sigma\*randn(1);

if X0(1)>=-3 & X0(1)<=12.1 & X0(2)>=4.1 & X0(2)<=5.8

Y(:,i+1)=X0;

i=i+1;

end

end

Temp=[X,Y];

W=zeros(1,2\*N);

for j=1:2\*N

p=0;

while p<N/2

k=1+fix((2\*N-1)\*rand());

if k~=j

p=p+1;

if f(Temp(:,j))>f(Temp(:,k))

W(j)=W(j)+1;

end

end

end

end

[W1,In]=sort(W);

I1=In(N+1:2\*N);

X=Temp(:,I1);

gen=gen+1;

maxW=f(Temp(:,In(end)));

if maxW>maxf

maxf=maxW;

opmx=Temp(:,In(end));

end

mean\_f(gen)=mean(W(I1));

fit(gen)=maxf;

end

opmx

maxf

plot(1:gen,fit,1:gen,mean\_f)