【算法设计】

# 

【程序代码】

C=unifrnd(-5,5,5,4);

C1=zeros(5,4);

EvalC=[0 0 0 0 0];

Weight=[0 0 0 0 0];

Roulette=[0 0 0 0 0];

Rand=[0 0 0 0 0];

Chromosome=[0 0 0 0 0];

Crossover=[0 0 0 0];

for i=1:5

EvalC(i)=1/(C(i,1)^2+C(i,2)^2+C(i,3)^2+C(i,4)^2+1);

end

EvalBest=max(EvalC);

a=1;

EvalBests(a)=EvalBest;

while(EvalBest<0.999)

Sum=sum(EvalC);

for i=1:5

Weight(i)=EvalC(i)/Sum;

end

Roulette(1)=Weight(1);

for i=2:5

Roulette(i)=Roulette(i-1)+Weight(i);

end

Rand=rand(1,5);

for i=1:5

if Rand(i)<Roulette(1)

Chromosome(i)=1;

elseif Rand(i)>Roulette(1) && Rand(i)<Roulette(2)

Chromosome(i)=2;

elseif Rand(i)>Roulette(2) && Rand(i)<Roulette(3)

Chromosome(i)=3;

elseif Rand(i)>Roulette(3) && Rand(i)<Roulette(4)

Chromosome(i)=4;

else

Chromosome(i)=5;

end

end

for i=1:5

C1(i,:)=C(Chromosome(i),:);

end

Rand=rand(1,5);

j=1;

for i=1:5

if Rand(i)<0.88

Crossover(j)=i;

j=j+1;

end

end

if Crossover(1)~=0 && Crossover(2)~=0

for i=unidrnd(4):4

t=C1(Crossover(1),i);

C1(Crossover(1),i)=C1(Crossover(2),i);

C1(Crossover(2),i)=t;

end

end

if Crossover(3)~=0 && Crossover(4)~=0

for i=unidrnd(4):4

t=C1(Crossover(3),i);

C1(Crossover(3),i)=C1(Crossover(4),i);

C1(Crossover(4),i)=t;

end

end

for i=1:5

for j=1:4

if rand()<0.1

C1(i,j)=10\*rand()-5;

end

end

end

for i=1:5

EvalC1(i)=1/(C1(i,1)^2+C1(i,2)^2+C1(i,3)^2+C1(i,4)^2+1);

end

EvalBest=max(EvalC1);

if EvalBest>EvalBests(a)

C=C1;

EvalC=EvalC1;

a=a+1;

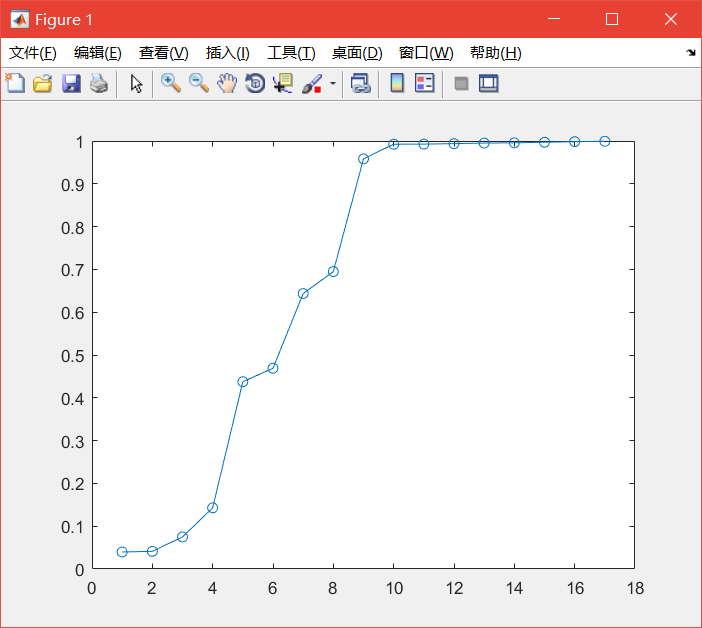
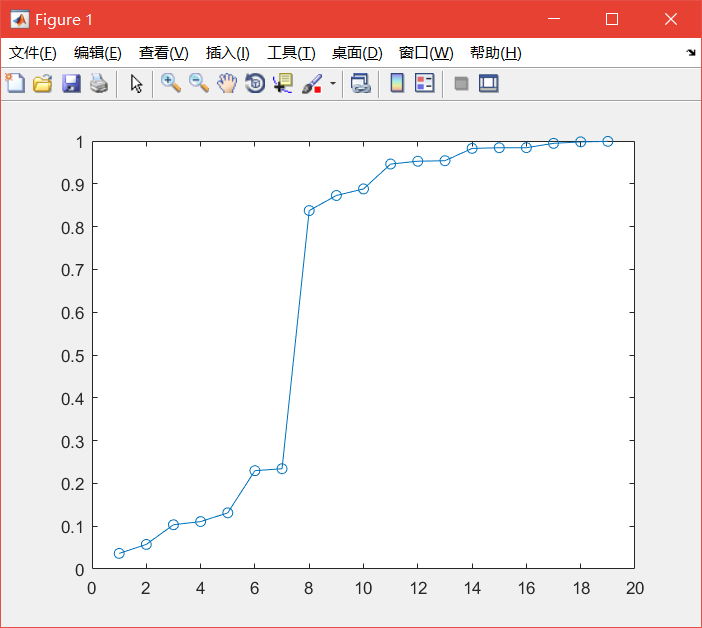
EvalBests(a)=EvalBest;

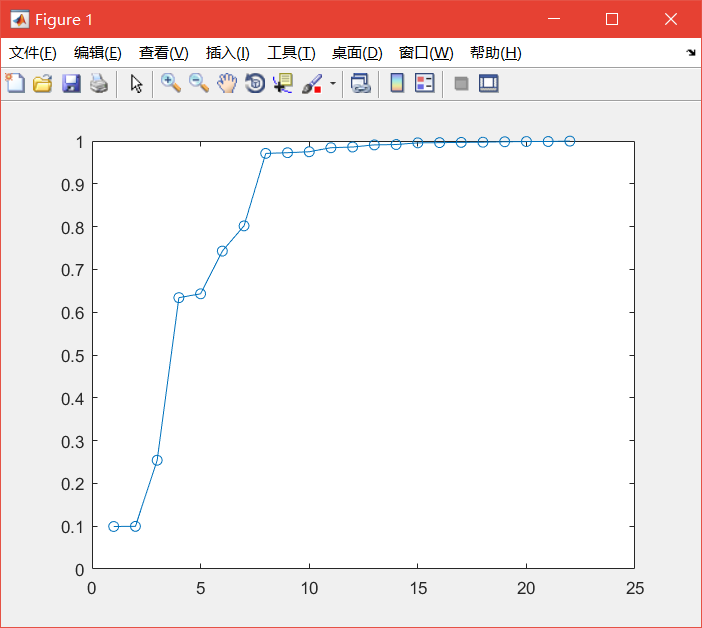
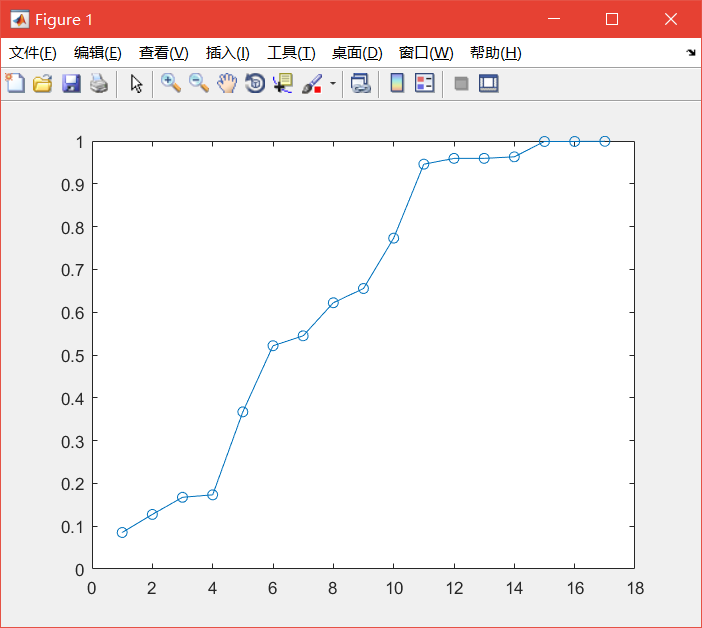
end

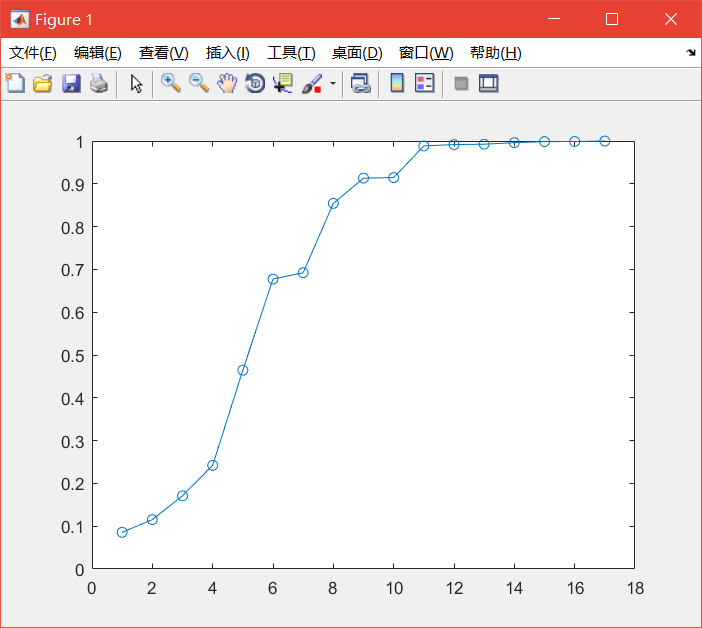
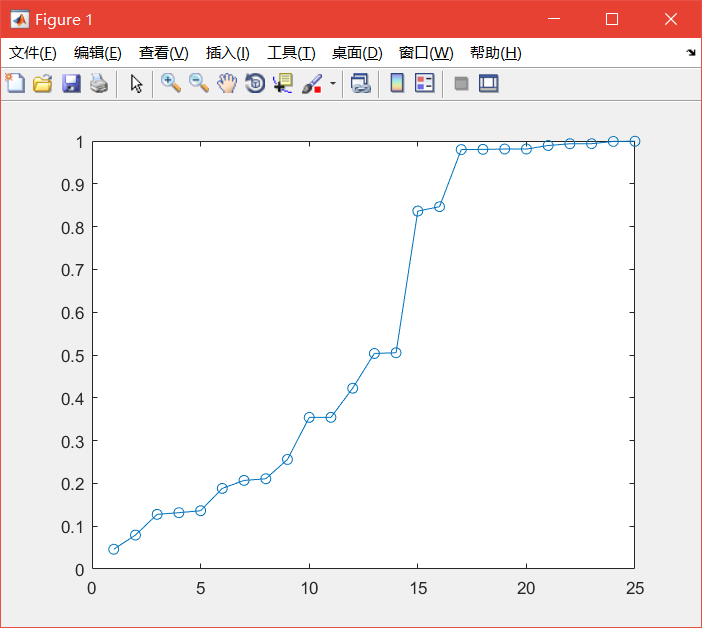
end

plot(EvalBests,'-o');

【实验结果】



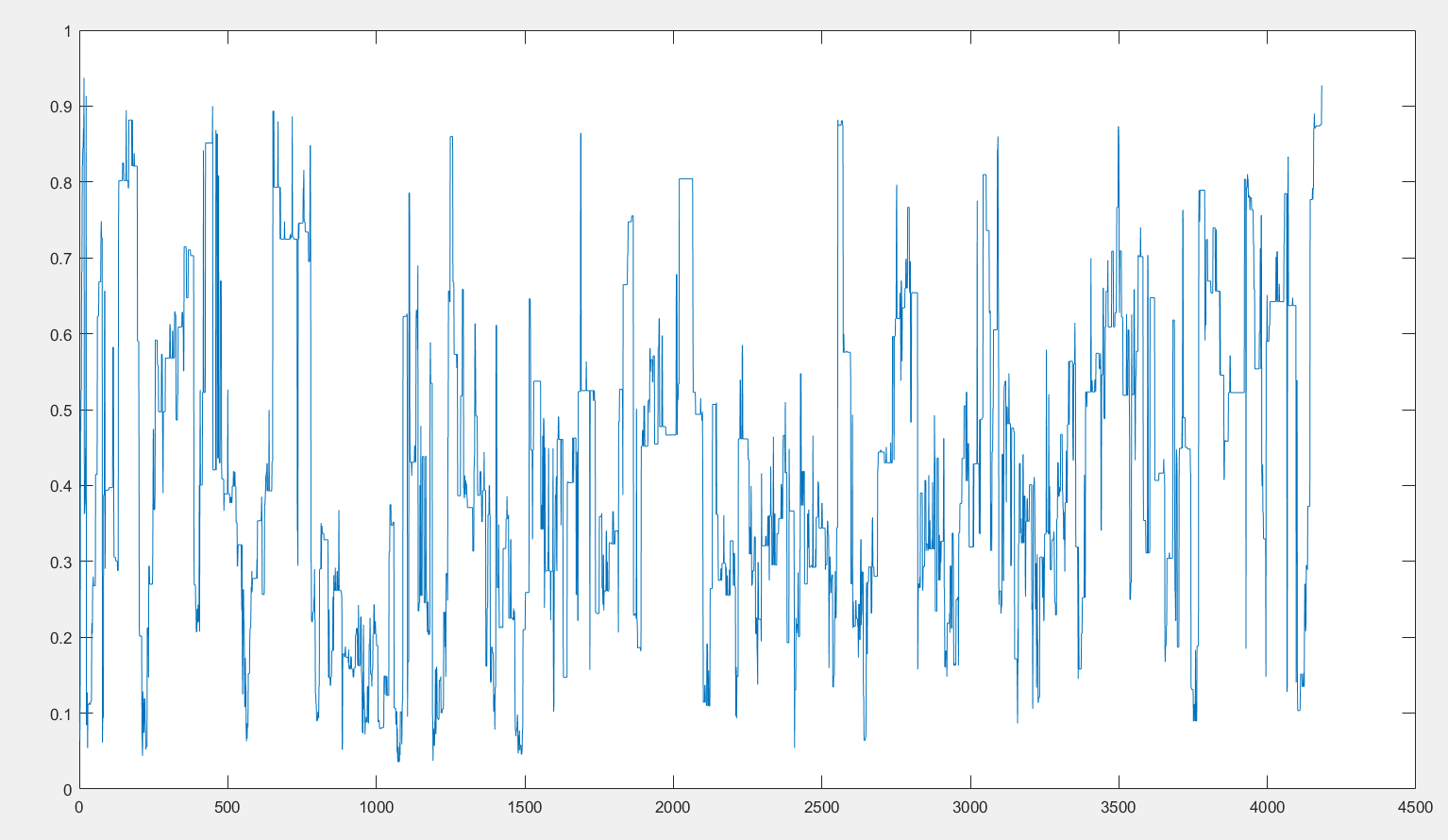




设置了终止条件为y值大于0.999，可以观察到都能在第25代前达到要求。

【实验总结】

这次实验开始进行的并不顺利，按照书上的步骤写完代码后发现最优解每一代的起伏很大，画图出来就像随机生成的一样，而且大概要几千代才能达到终止条件。



书上的Step6 变异后的新群体取代原有群体，重新计算群体中各个染色体的适应值。倘若群体的最大适应值大于Best的适应值，则以该最大适应值对应的染色体替代Best。

我将这一步的的变异后的新群体取代原有群体放到后面进行，倘若群体的最大适应值大于Best的适应值，则变异后的新群体取代原有群体，否则不取代，让原来的种群重新选择交配变异，这样做的目的是防止退化，保证每一代的种群都比上一代好。