实验10 免疫算法实验

姓名：唐川淇 学号：1131190111

# 问题重述

使用免疫算法求解下面函数的极值：

# 问题分析

免疫算法是模仿生物免疫机制，结合基因的进化机理，人工构造出的一种新型智能优化算法。采用群体搜索策略，通过迭代计算，最终以较大的概率得到问题的最优解。相比于其他算法，免疫算法利用自身产生多样性和维持机制的特点，保证了群体多样性，克服了‘早熟’问题，可以得到全局最优解。具有自适应性，随机性并行性，全局收敛性，种群多样性等特点。

免疫算法中其中定义的变量和问题有如下的对应关系：

|  |  |
| --- | --- |
| 生物名词 | 问题元素 |
| 抗原 | 优化问题 |
| 抗体 | 优化问题的可行解 |
| 亲和度 | 可行解的质量 |
| 细胞活化 | 免疫选择 |
| 细胞分化 | 个体克隆 |
| 亲和度成熟 | 变异 |
| 动态维持平衡 | 种群刷新 |

免疫算法有如下的特点：

1. 全局搜索能力，免疫算法在对优质抗体邻域进行局部搜索的同时利用变异算子和种群刷新不断产生新个体，探索可行解的新区域，保证在完整的可行解区间进行搜索。
2. 多样性保持机制。对抗体的浓度进行计算，将结果作为评价抗体个体优劣的重要标准。
3. 鲁棒性强。对于物体的初始解依赖不强。
4. 并行分布式搜索机制。

# 模型的建立与求解

## 模型公式

### 亲和度评价算子

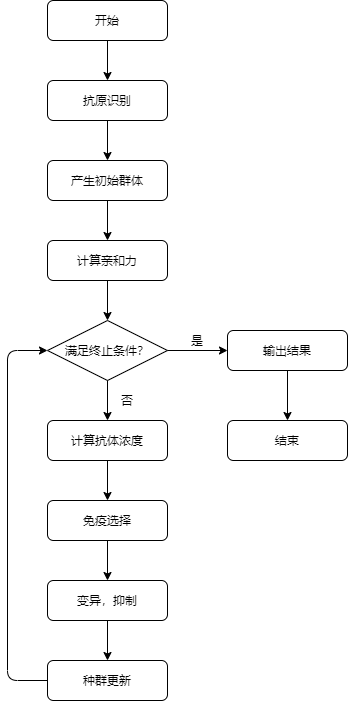
亲和度评价算子表征免疫细胞与抗原的结合程度，定义为函数aff(x)，其中s代表的是问题的可行解区间，函数的输入作为一个抗体。亲和度用抗体向量间的欧氏距离来计算。

### 抗体浓度评价算子

抗体浓度算子定义为：

S函数代表的是抗体之间的相似程度，若小于某一个阈值则为1，若大于阈值则为0，式中的N为种群规模。

算法流程图如下：



## 模型建立

### 初始化

首先根据题目初始化参数，代码如下：

|  |
| --- |
| 代码 |
| 介绍：初始化参数 |
| D=5; %免疫个体维数  NP=50; %免疫个体数目  Xs=10; %取值上限  Xx=-10; %取值下限  G=30; %最大免疫代数  pm=0.7; %变异概率  alfa=1; %激励度系数  belta=1; %激励度系数  detas=0.2; %相似度阈值  gen=0; %免疫代数  Nc1=10; %克隆个数  deta0=1\*Xs; %邻域范围初值 |

其中个体维数设定为5，免疫个体数目设定为50，上下限按照题目要求分别设定为10和-10，由于问题求解的函数比较简单，设定最大免疫代数为30。

初始化参数之后，使用随机函数rand()初始化种群。代码如下：

|  |
| --- |
| 代码 |
| 介绍：初始化种群 |
| f= rand(D,NP)\*(Xs-Xx)+Xx;  for np=1:NP  MSLL(np)=func1(f(:,np));  end |

初始化种群之后，首先计算出之后迭代需要用到的参数，包括个体浓度等，之后按照激励度升序排列，代码如下：

|  |
| --- |
| 代码 |
| 介绍：计算参数 |
| %计算个体浓度和激励度  for np=1:NP  for j=1:NP  nd(j)=sum(sqrt((f(:,np)-f(:,j)).^2)); %亲和度，欧氏距离  if nd(j)<detas  nd(j)=1;  else  nd(j)=0;  end  end  ND(np)=sum(nd)/NP; %抗体浓度  end  MSLL=alfa\*MSLL-belta\*ND; %抗体激励度  %激励度按升序排列  [SortMSLL,Index]=sort(MSLL);  Sortf=f(:,Index); |

### 免疫循环

首先选择激励度前NP/2的个体，代码如下：

|  |
| --- |
| 代码 |
| 介绍：选择个体 |
| a=Sortf(:,i);  Na=repmat(a,1,Nc1);  deta=deta0/gen; |

进行免疫操作，同时对超越边界的数据进行处理，代码如下：

|  |
| --- |
| 代码 |
| 介绍：免疫操作 |
| for j=1:Nc1  for ii=1:D  %变异  if rand<pm  Na(ii,j)=Na(ii,j)+(rand-0.5)\*deta;  end  %边界条件处理  if (Na(ii,j)>Xs)||(Na(ii,j)<Xx)  Na(ii,j)=rand\*(Xs-Xx)+Xx;  end  end  end  Na(:,1)=Sortf(:,i); |

保留个体中亲和度最高的个体，代码如下：

|  |
| --- |
| 代码 |
| 介绍：克隆抑制 |
| for j=1:Nc1  NaMSLL(j)=func1(Na(:,j));  end  [NaSortMSLL,Index]=sort(NaMSLL);  aMSLL(i)=NaSortMSLL(1);  NaSortf=Na(:,Index);  af(:,i)=NaSortf(:,1); |

计算免疫种群的激励度：

|  |
| --- |
| 代码 |
| 介绍：计算免疫种群的激励度 |
| for np=1:NP/2  for j=1:NP/2  nda(j)=sum(sqrt((af(:,np)-af(:,j)).^2));  if nda(j)<detas  nda(j)=1;  else  nda(j)=0;  end  end  aND(np)=sum(nda)/NP/2;  end |

种群刷新：

|  |
| --- |
| 代码 |
| 介绍：种群刷新 |
| bf=rand(D,NP)\*(Xs-Xx)+Xx;  for np=1:NP/2  bMSLL=func1(bf(:,np));  end |

新种群的激励度：

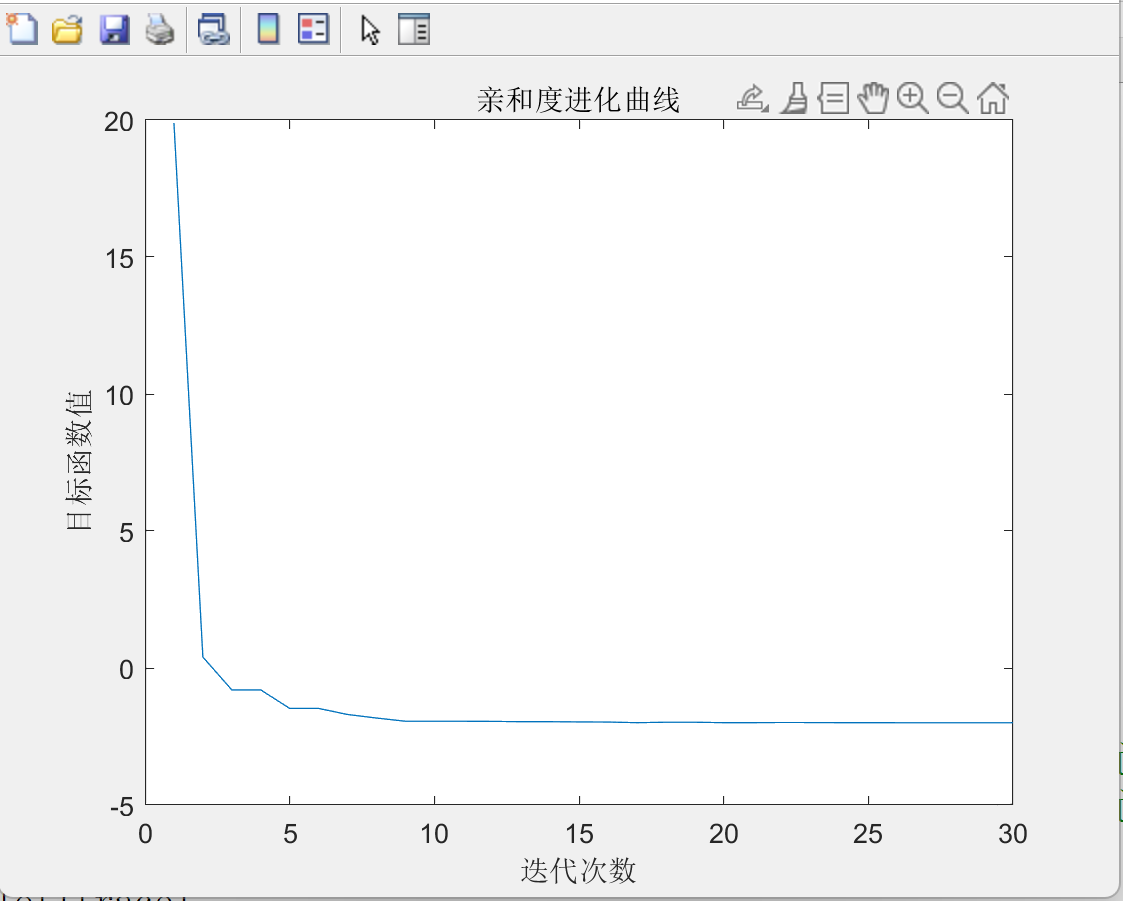
|  |
| --- |
| 代码 |
| 介绍：新种群的激励度 |
| for np=1:NP/2  for j=1:NP/2  ndc(j)=sum(sqrt((bf(:,np)-bf(:,j)).^2));  if ndc(j)<detas  ndc(j)=1;  else  ndc(j)=0;  end  end  bND=sum(ndc)/NP/2;  end |

免疫种群和新种群合并：

|  |
| --- |
| 代码 |
| 介绍：免疫种群和新种群合并 |
| f1=[af,bf];  MSLL=[aMSLL,bMSLL];  [SortMSLL,Index]=sort(MSLL);  Sortf=f1(:,Index);  gen=gen+1;  trace(gen)=func1(Sortf(:,1)); |

## 结果

亲和度曲线变化如下：



最终能够得到的最优个体是：

-0.00642972422456145

0.00537005675119565

-0.00699548977346456

0.0318272803306463

-0.0275708311178786

附录

|  |
| --- |
| 代码 |
| 介绍：完整代码 |
| clear all;  close all;  clc;  D=5; %免疫个体维数  NP=50; %免疫个体数目  Xs=10; %取值上限  Xx=-10; %取值下限  G=30; %最大免疫代数  pm=0.7; %变异概率  alfa=1; %激励度系数  belta=1; %激励度系数  detas=0.2; %相似度阈值  gen=0; %免疫代数  Nc1=10; %克隆个数  deta0=1\*Xs; %邻域范围初值  %初始种群  f= rand(D,NP)\*(Xs-Xx)+Xx;  for np=1:NP  MSLL(np)=func1(f(:,np));  end  %计算个体浓度和激励度  for np=1:NP  for j=1:NP  nd(j)=sum(sqrt((f(:,np)-f(:,j)).^2)); %亲和度，欧氏距离  if nd(j)<detas  nd(j)=1;  else  nd(j)=0;  end  end  ND(np)=sum(nd)/NP; %抗体浓度  end  MSLL=alfa\*MSLL-belta\*ND; %抗体激励度  %激励度按升序排列  [SortMSLL,Index]=sort(MSLL);  Sortf=f(:,Index);  %免疫循环  while gen<G  for i=1:NP/2  %选择激励度为前NP/2的个体进行免疫操作  a=Sortf(:,i);  Na=repmat(a,1,Nc1);  deta=deta0/gen;  for j=1:Nc1  for ii=1:D  %变异  if rand<pm  Na(ii,j)=Na(ii,j)+(rand-0.5)\*deta;  end  %边界条件处理  if (Na(ii,j)>Xs)||(Na(ii,j)<Xx)  Na(ii,j)=rand\*(Xs-Xx)+Xx;  end  end  end  Na(:,1)=Sortf(:,i); %保留克源个体  %克隆抑制，保留亲和度最高个体  for j=1:Nc1  NaMSLL(j)=func1(Na(:,j));  end  [NaSortMSLL,Index]=sort(NaMSLL);  aMSLL(i)=NaSortMSLL(1);  NaSortf=Na(:,Index);  af(:,i)=NaSortf(:,1);  end  %免疫种群激励度  for np=1:NP/2  for j=1:NP/2  nda(j)=sum(sqrt((af(:,np)-af(:,j)).^2));  if nda(j)<detas  nda(j)=1;  else  nda(j)=0;  end  end  aND(np)=sum(nda)/NP/2;  end  aMSLL=alfa\*aMSLL-belta\*aND;  %种群刷新  bf=rand(D,NP)\*(Xs-Xx)+Xx;  for np=1:NP/2  bMSLL=func1(bf(:,np));  end  %新生成种群激励度  for np=1:NP/2  for j=1:NP/2  ndc(j)=sum(sqrt((bf(:,np)-bf(:,j)).^2));  if ndc(j)<detas  ndc(j)=1;  else  ndc(j)=0;  end  end  bND=sum(ndc)/NP/2;  end  bMSLL=alfa\*bMSLL-belta\*bND;  %免疫种群与新种群合并  f1=[af,bf];  MSLL=[aMSLL,bMSLL];  [SortMSLL,Index]=sort(MSLL);  Sortf=f1(:,Index);  gen=gen+1;  trace(gen)=func1(Sortf(:,1));  end  Bestf=Sortf(:,1); %最优变量  trace(end); %最优值  figure  plot(trace)  xlabel('迭代次数')  ylabel('目标函数值')  title('亲和度进化曲线')  %亲和度函数  function value=func1(x)  value=sum(x.^2)-2;  end |