统计推断在数模、模数转换系统中的应用

组号：43　姓名：于哲昆(组长) 学号：5130309547　姓名：姚若晨　学号：5130309565

摘要：本文介绍了遗传算法的理论基础及工作原理。作者通过遗传算法并结合统计数学的方法，为某型产品内部的一个监测模块，寻求校准工序的优化方案。遗传算法对表达式没有特殊要求、能高效地随机搜索优化方案、全局性和可操作性比较强，故其广为应用。

关键词：遗传算法，定标

ABSTRACT： Article introduces the theory of Genetic Algorithm and how it works. The writers search for a method for a calibration about a monitor module in a kind of product according to GA together with statistics mathematics method. The GA doesn't have special require for expression and advantages so on make itself a well-applied way to solve actual problems.

Keywords: Genetic Algorithm, calibration.

1. 引言

在工业生产中，需要对元器件的特性进行标定，即对一系列的测试数据进行分析，找出足以反映特性的规律。但是用一种元器件在大规模生产中，有可能元器件的个体差异很大。若要对每一个元器件进行完全精确的标定显然费时费力，不符合工业化生产的初衷，没有效率可言。那么，找出一种实际可行的方法，尽量简化标定过程，降低时间和资金成本就显得很必要。比如，在寻求测试数据的函数关系上，是不是可以通过测试尽可能少的点来得到尽可能准确的标定结果（误差允许范围内，不影响元器件的使用）。

2. 研究目的和背景

假定有某型投入批量试生产的电子产品，其内部有一个模块，功能是监测某项与外部环境有关的物理量（可能是温度、压力、光强等）。该监测模块中传感器部件的输入输出特性呈明显的非线性。本课题要求为该模块的批量生产寻求校准工序的优化方案设计一种成本合理的传感特性校准（定标工序）方案。

2.1 成本问题：（1）测定点数量越多，测定成本越高

（2）估算点数量越多，误差损失（成本）越高

2.2 降低定标成本：尽量减少测定的次数用测定的点的数据，推断估算出其他点的数值。

方法一：线性插值。

方法二：非线性插值。

步骤：第 1 步：拟合——利用已测定点的数据，确定一条反映连续变化趋势的曲线

第 2 步：插值——利用这条曲线，确定未知点数据

2.3 数学模型

为了对本课题展开有效讨论，需建立一个数学模型，对问题的某些方面进行必要的描述和限定。

监测模块的组成框图如图 1。其中，传感器部件（包含传感器元件及必要的放大电路、

调理电路等）的特性是我们关注的重点。传感器部件监测的对象物理量以符号 Y 表示；传感部件的输出电压信号用符号 X 表示，该电压经模数转换器（ADC）成为数字编码，并能被微处理器程序所读取和处理，获得信号 $\hat{Y}$ 作为 Y 的读数（监测模块对 Y 的估测值）。

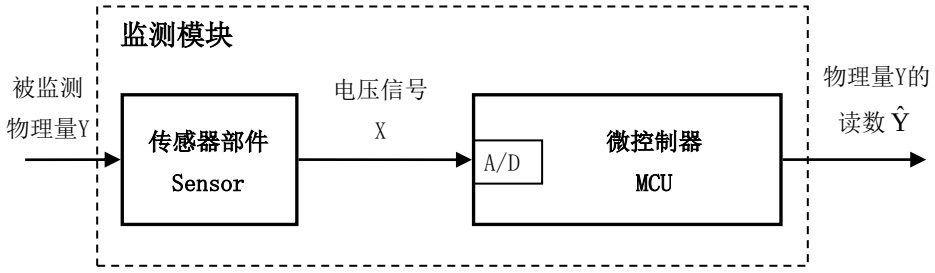


图 1 监测模块组成框图

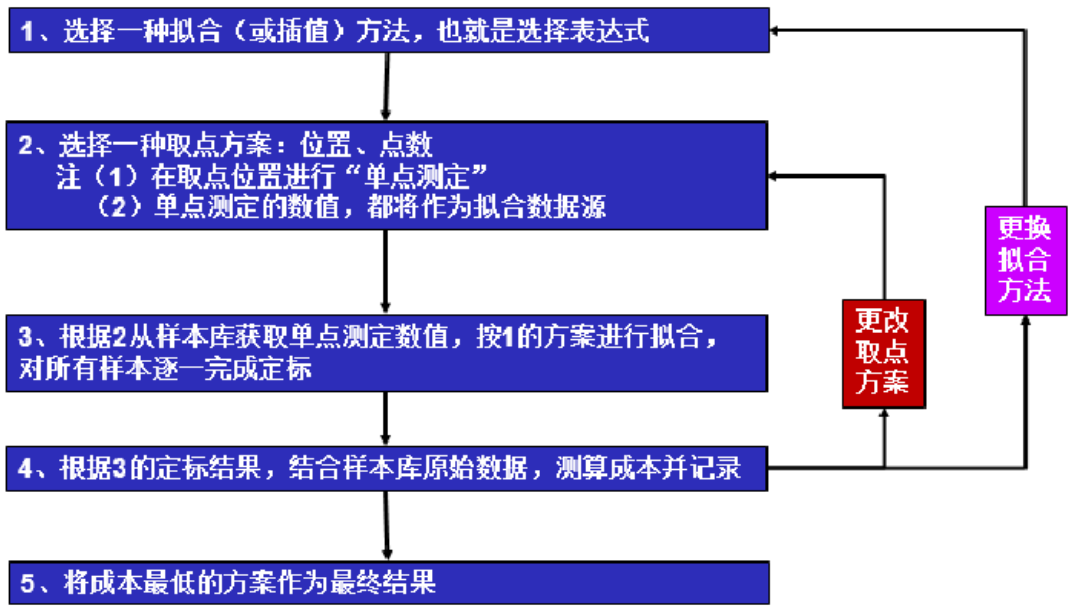
所谓传感特性校准，就是针对某一特定传感部件个体，通过有限次测定，估计其 Y 值与 X 值间一一对应的特性关系的过程。数学上可认为是确定适用于该个体的估测函数 $\hat{y}=f(x)$ 的过程，其中 x 是 X 的取值，$\hat{y}$ 是对应 Y 的估测值。

考虑实际工程中该监测模块的应用需求，同时为便于在本课题中开展讨论，我们将问题限于 X 为离散取值的情况，规定

$$X \in \{x_1, x_2, x_3, ..., x_{50}, x_{51}\} = \{5.0, 5.1, 5.2, ..., 9.9, 10.0\}$$

相应的 Y 估测值记为 $\hat{y}_i = f(x_i)$，Y 实测值记为 y_i，$i = 1, 2, 3, ..., 50, 51$。

2.4 求解路径：

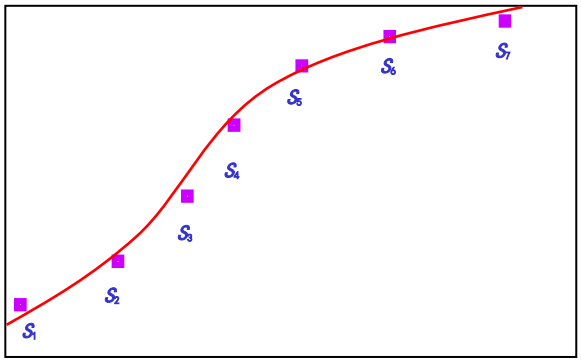


3．拟合方法的讨论：

3.1 三次多项式拟合三次多项式拟合

1．使用三次多项式对选取的七个特征点组合即可能解进行拟合。样本为 S={S1,S2,…S7}。

$$u = a_1 d^3 + a_2 d^2 + a_3 d + a_4 \qquad (4\text{-}1)$$

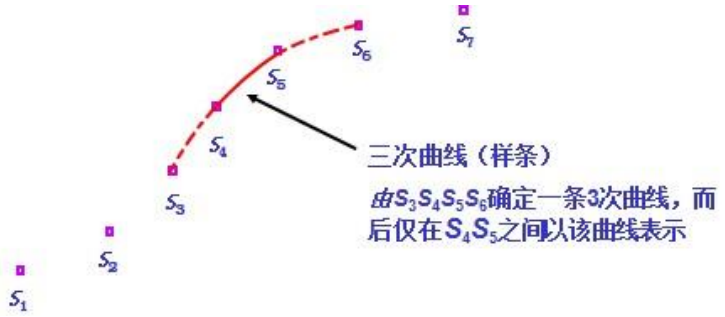


2．曲线的拟合度。将去掉 1 和 51 号点的其余 49 个点的输出值代入上述三次多项式计算出相应个输出值，通过评价函数得到评价分值。

3．实现最优化过程，即不断的寻找存在的可能解，继续前两步骤。每次得到可能解，通过评价，做出一定概率的接受或者舍弃，实现当前解的优化，直至达到终止条件。

3.2 三次样条插值拟合法

1．使用三次样条插值对确定一个可能解，即 7 个特征点组合 S={s1,s2,…,s7}进行拟合。具体方法为：首先对于非两端点，以四个连续点确定一条三次曲线，但仅在中间两点之间用该三次曲线表示，以此类推，所有非两端点之间均有三次曲线。两端点由端点处三个点用二次曲线拟合。



三次样条插值曲线拟合

2．估该曲线的拟合度。将去掉 1 和 51 号点的其余 49 个点的 X 代入得到三次样条插值后相应的 49 个 Y，通过评价函数得到评价分值。

3． 不断的寻找存在的可能解，继续前两步骤。每次得到可能解，通过评价，做出一定概率的接受或者舍弃，实现当前解的优化，直至达到终止条件。

4．算法的选取：

4.1 穷举法

数学中，很多问题都涉及到寻求最优解。有的问题，可行解数目较少，比如只有 2^3=8 中可行解。此时，只需要逐一列举，并进行比对就可以得出结论找出最优解。但是对于本实验，如何使用穷举法，由于数据过多，计算机必须要完成 C_{53} 次拟合，才能找出最优解。如此巨大而数目，要求解多项式，即使是计算机处理起来也将费时费力，甚至无法完成。因此，必须要选取更加合理高效的算法，使用统计知识求解。

4.2 模拟退火算法

方法介绍：用固体退火模拟组合优化问题，将内能 E 模拟为目标函数值 f，温度 T 演化成控制参数 t,即得到解组合优化问题的模拟退火算法:由初始解 i 和控制参数初值 t 开始，对当前解重复“产生新解→计算目标函数差→接受或舍弃”的迭代，并逐步衰减 t 值，算法终止时的当前解即为所得近似最优解。

优缺点分析：使用 matlab 编程，相比有遗传算法，代码易写，计算时间也较短。实际执行中，结果具有一定的偶然性。

4.3 遗传算法

遗传算法（Genetic Algorithm）是一类借鉴生物界的进化规律（适者生存，优胜劣汰遗传机制）演化而来的随机化搜索方法。其主要特点是直接对结构对象进行操作，不存在求导和函数连续性的限定；具有内在的隐并行性和更好的全局寻优能力；采用概率化的寻优方法，能自动获取和指导优化的搜索空间，自适应地调整搜索方向，不需要确定的规则。

算法实现过程：

a)初始化：设置进化代数计数器 t=0，设置最大进化代数 T，随机生成 M 个个体作为初始群体 P(0)。

b)个体评价：计算群体 P(t)中各个个体的适应度。

c)选择运算:将选择算子作用于群体。选择的目的是把优化的个体直接遗传到下一代或通过配对交叉产生新的个体再遗传到下一代。选择操作是建立在群体中个体的适应度评估基础上的。

d)交叉运算：指把两个父代个体的部分结构加以替换重组而生成新个体的操作。

e)变异运算：将变异算子作用于群体。即是对群体中的个体串的某些基因座上的基因值作变动。群体 P(t)经过选择、交叉、变异运算之后得到下一代群体 P(t+1)。

可以有以下的算法：a)实值变异 b)二进制变异。

一般来说，变异算子操作的基本步骤如下：a)对群中所有个体以事先设定的变异概率判断是否进行变异。b)对进行变异的个体随机选择变异位进行变异。

f)终止条件判断:若 t=T,则以进化过程中所得到的具有最大适应度个体作为最优解输出，终止计算。

优缺点分析：此方法借鉴生物进化理论，通过选择，交叉配对，变异和淘汰，可以得到最优解。并且，相比于穷举法，大大减少了计算时间。

遗传算法的一般算法：

a）建初始状态：初始种群是从解中随机选择出来的，将这些解比喻为染色体或基因，该种群被称为第一代，这和符号人工智能系统的情况不一样，在那里问题的初始状态已经给定了。

b）评估适应度：对每一个解(染色体)指定一个适应度的值，根据问题求解的实际接近程度来指定(以便逼近求解问题的答案)。不要把这些“解”与问题的“答案”混为一谈，可以把它理解成为要得到答案，系统可能需要利用的那些特性。

c）繁殖（包括子代突变）：带有较高适应度值的那些染色体更可能产生后代(后代产生后也将发生突变)。后代是父母的产物，他们由来自父母的基因结合而成，这个过程被称为“杂交”。

d）下一代：如果新的一代包含一个解，能产生一个充分接近或等于期望答案的输出，那么问题就已经解决了。如果情况并非如此，新的一代将重复他们父母所进行的繁衍过程，一代一代演化下去，直到达到期望的解为止。

e）并行计算：非常容易将遗传算法用到并行计算和群集环境中。一种方法是直接把每个节点当成一个并行的种群看待。然后有机体根据不同的繁殖方法从一个节点迁移到另一个节点。另一种方法是“农场主/劳工”体系结构，指定一个节点为“农场主”节点，负责选择有机体和分派适应度的值，另外的节点作为“劳工”节点，负责重新组合、变异和适应度函数的评估。

最终选取遗传算法。

5.解决问题

确定具体方法：通过分析比对，我们确定对每组 51 个数据，只选取其中的 7 个点进行拟合并计算适用度。每组数据选取的七个点的下标一致。考虑到难易程度，时间和效率的问题，我们最终确定使用遗传算法。

5.1 确定方法实现

（1）读入老师提供的 469 组数据。

（2）随机函数随机排序并选取前七个点，再次排序，得到随机的几个点。

（3）用遗传算法作为大循环。

（4）在循环内前部分加入随机变化一个点生成新的七个点组合。并用拟合法得到每个 y(i,j)-y’(i,j)的值。

（5）循环中间部分加入计算去掉两个端点的 49 个点的评价函数的分值，并计算 469 组数据的平均评价分值。

（6）循环的后半部分决定是否接受这七个点。若分值大于最大分值，则接受。

（7）输出最优解。

5.2 成本计算

为评估和比较不同的校准方案，特制定以下成本计算规则。

- 单点定标误差成本

$$s_{i,j}=\begin{cases}0 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|\le 0.5\\ 0.5 & \text{if } 0.5<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 1\\ 1.5 & \text{if } 1<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 2\\ 6 & \text{if } 2<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 3\\ 12 & \text{if } 3<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 5\\ 25 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|>5\end{cases} \qquad (1)$$

单点定标误差的成本按式（1）计算，其中 $y_{i,j}$ 表示第 i 个样本之第 j 点 Y 的实测值，$\hat{y}_{i,j}$ 表示定标后得到的估测值（读数），该点的相应误差成本以符号 $s_{i,j}$ 记。

- 单点测定成本

实施一次单点测定的成本以符号 q 记。本课题指定 q=20。

- 某一样本个体的定标成本

$$S_i = \sum_{j=1}^{51} s_{i,j} + q \cdot n_i \quad (2)$$

对样本 i 总的定标成本按式（2）计算，式中 n_i 表示对该样本个体定标过程中的单点测定次数。

- 校准方案总体成本

按式（3）计算评估校准方案的总体成本，即使用该校准方案对标准样本库中每个样本个体逐一定标，取所有样本个体的定标成本的统计平均。

$$C = \frac{1}{M} \sum_{i=1}^{M} S_i \quad (3)$$

总体成本较低的校准方案，认定为较优方案。

5.3 交叉的实现

在选择出新种群父代后，就在父代中进行交叉。本例采用单点交叉即在基因段中从随机一点断裂，并与另一基因进行重组。为了使基因充分融合，将交叉概率定为 0.9。通过生成 0 到 1 的随机数，若小于 0.9 则进行交叉来实现。将父代随机排列，并进行编号，按照 i 和（pop_size-i+2）的规则结对。若判定二者进行交叉，则随机产生一个基因断裂点，将二者基因重组后输入新的种群。若判定二者不进行交叉，则直接将二者输入新的种群。

5.4 变异的实现

变异就是指将基因中 0、1 进行互换。首先我们给定基因变异的概率，因为变异概率只是提供多样化的选择，不能因为变异而改变基因整体的优化性，因此变异的概率应该较小，定为 0.01。而种群中每一个个体的每一个基因均有可能发生突变。鉴于种群是用二维矩阵表示，所以变异的列举应该相应地使用二重循环进行实现。

6 结果分析

6.1 程序运行结果图示（100 次）

postion : [1 9 18 26 35 44 51] mean_cost: 95.808102

position: [1 10 20 28 35 44 51] mean_cost: 95.487207

position: [1 8 20 27 29 36 45 51] mean_cost: 104.024520

position: [1 5 20 26 34 44 51] mean_cost: 97.455224

position: [1 10 21 28 37 46 51] mean_cost: 97.037313

position: [1 7 16 25 32 36 42 47 51] mean_cost: 114.272921

position: [1 9 20 22 27 32 43 51] mean_cost: 105.646055

position: [1 7 9 15 27 35 45 51] mean_cost: 110.820896

position: [1 12 24 30 43 51] mean_cost: 100.796375

position: [1 11 20 26 34 45 51] mean_cost: 96.640725

position: [1 7 20 30 36 45 51] mean_cost: 98.619403

position: [1 6 20 29 36 45 51] mean_cost: 97.970149

position: [1 12 16 27 31 44 51] mean_cost: 103.994670

position: [1 11 23 31 43 51] mean_cost: 96.866738

position: [1 10 21 25 29 42 46 51] mean_cost: 107.875267

position: [1 7 18 24 33 44 51] mean_cost: 97.140725

position: [1 7 22 30 40 51] mean_cost: 100.263326

position: [1 10 21 29 35 45 51] mean_cost: 96.459488

position: [1 5 15 23 33 42 44 51] mean_cost: 107.753731

position: [1 11 21 29 36 46 51] mean_cost: 97.325160

position: [1 5 19 26 31 41 46 51] mean_cost: 105.598081

position: [1 8 9 20 27 35 44 51] mean_cost: 106.487207

position: [1 2 12 20 27 34 46 51] mean_cost: 106.377399

position: [1 8 20 26 44 51] mean_cost: 118.348614

position: [1 9 21 27 34 44 51] mean_cost: 95.524520

position: [1 9 21 31 42 51] mean_cost: 96.092751

position: [1 7 20 27 35 44 51] mean_cost: 95.715352

position: [1 10 20 29 41 51] mean_cost: 97.245203

position: [1 9 20 26 33 44 51] mean_cost: 95.311301

position: [1 10 22 28 37 47 51] mean_cost: 98.265458

position: [1 7 17 27 35 44 51] mean_cost: 96.754797

position: [1 10 20 26 35 44 51] mean_cost: 95.605544

position: [1 8 19 26 34 45 51] mean_cost: 95.921109

position: [1 9 21 28 35 43 51] mean_cost: 95.973348

position: [1 11 22 32 44 51] mean_cost: 96.266525

position: [1 12 21 25 33 44 51] mean_cost: 98.062900

position: [1 8 16 27 34 44 51] mean_cost: 97.816631

position: [1 10 20 26 35 44 51] mean_cost: 95.605544

position: [1 10 20 26 35 44 51] mean_cost: 95.605544

position: [1 8 19 26 34 45 51] mean_cost: 95.921109

position: [1 10 20 26 35 44 51] mean_cost: 95.605544

position: [1 10 20 26 35 44 51] mean_cost: 95.605544

position: [1 10 20 26 35 44 51] mean_cost: 95.605544

position: [1 10 20 26 35 44 51] mean_cost: 95.605544

6.2 结果分析

在 90 次左右趋于稳定。

因此，较优的校准方案是：在 1,10,20,26,35,44,51 处定标较好，成本为 95.605544.

7 拓展部分

与其他算法的对比

启发式搜索算法除了遗传算法外，还有其他算法。现在我们将遗传算法与另一种应用广泛的启发式搜索算法模拟退火算法进行比较。

模拟退火算法由自然界中固体由高温时内部粒子的无序状态向低温时内部粒子向有序状态进行变化的事实得到启发。根据 Metropolis 准则，粒子在温度 T 时趋于平衡的概率为 e-ΔE/(kT).其中，E 是温度 T 时的内能，ΔE 为内能改变量，k 为 Boltzmann 常数。模拟退火即模拟该过程。在退火算法中，用内能 E 来模拟目标函数，温度 T 为控制参数 t。由 t 和一个初始解开始，对当前解不断产生新解，计算目标函数差，接受或舍弃新解的迭代步骤，同时每一步迭代时减小 t 的值，最终当温度降低到特定值时，算法结束，得到近似的最优解。

遗传算法主要由选择，交叉，变异等操作组成，通过种群进行进化。两者的不同主要体现在选择的方式上，模拟退火是采用单个个体进行进化，不断优化单个答案，虽然精度较高但是时间较长。遗传算法是采用种群进行进化。模拟退火一般新解优于当前解才接受新解，并且还需要通过温度参数 t 进行选择，并通过变异操作产生新个体。而遗传算法新解是通过选择操作进行选择个体，并通过交叉和变异产生新个体。主要通过种群的进化进行最优解的选择，虽然得不到最好的最优解，但可以得到一个广阔的近似解空间，使解的选择性较多，且时间较短。相同点是都采用进化控制优化的过程。

8 参考文献

[1] 袁炎：统计推断讲座
[2] 网络资源 China-pub.com 《matlab 教程》
[3] 网络资源 http://baike.baidu.com/view/18185.htm

附录

```
%主程序
data = csvread('20141010dataform.csv');
pop_size = 100;
cross_rate=0.9;交叉概率
mutation_rate=0.01;%突变概率
generation_size=100;%进化代数
Y=zeros(469,51);
Y(1:469,:)=data(2:2:938,:);
gene=geneinit(pop_size);
for G=1:generation_size
    display(G);
    cost=fitness(gene,Y,pop_size);
    gene=select(gene,cost,pop_size);
    gene=generate(gene,pop_size,cross_rate);
    gene=mutate(gene,pop_size,mutation_rate);
    display(find(gene(1,:)==1));
end
xx=find(gene(1,:)==1);
evaluate(xx,Y);计算平均成本


function out = geneinit(pop_size)%随机产生初始种群
out=round(rand(pop_size,51)-0.2);
out(:,1)=1;
out(:,51)=1;
end

function out =fitness (gene,Y,pop_size)%计算每个个体平均成本
out=zeros(pop_size,1);
X=5:0.1:10;
for i=1:pop_size
    c=sum(gene(i,:)==1);%测试点数量
    pos=find(gene(i,:)==1);%测试点位置
    xx=5+(pos-1)*0.1;%测试点 X 值
    yy=Y(:,pos);%测试点 Y 值
    f=spline(xx,yy);
    difference=ppval(f,X)-Y;%理论值与实际值的差
    out(i)=12*c+errorcost(difference)/469;%单个个体平均成本
end
min(out)
mean(out)
end
```

```
function out = select(gene,cost,pop_size)%选择
out=zeros(pop_size,51);
cost0=max(cost)-cost;
s0=sum(cost0);
s=zeros(pop_size+1);
s(1)=0;
s(pop_size+1)=1;
s(2:pop_size)=sum(cost0(1:pop_size-1))/s0;
for i=2:pop_size
    t=rand();
    j=search(t,s,1,pop_size+1);
    out(i,:)=gene(j,:);
end
sort0=[[1:pop_size]',cost];
sort0=sortrows(sort0,2);
out(1,:)=gene(sort0(1,1),:);
end


function [out] = search(in,s,l,r)%查找
%使用二分法
mid=floor((l+r)/2);
if in<=s(mid)
    if in>s(mid-1)
        out=mid-1;
    else
        out=search(in,s,l,mid);
    end
else
    if in<=s(mid+1)
        out=mid;
    else
        out=search(in,s,mid,r);
    end
end
end


function out = mutate(gene,pop_size,mutation_rate)%变异
out=gene;
for i=2:pop_size;
    for j=2:50;
```

```
            t=rand();
            if t<=mutation_rate
                  out(i,j)=~out(i,j);
            end
      end
end
end


function out = generate(gene,pop_size,cross_rate)%交叉
for i=2:floor(pop_size/2+1)
      out=gene;
      mid=floor(rand()*50)+1;
      t=rand();
      if t<=cross_rate
            out(i,1:mid)=gene(pop_size-i+2,1:mid);
            out(pop_size-i+2,1:mid)=gene(i,1:mid);
            out(i,mid+1:51)=gene(pop_size-i+2,mid+1:51);
            out(pop_size-i+2,mid+1:51)=gene(i,mid+1:51);
      end
end
end


function [out] = errorcost(diffrernce)%单个个体平均误差成本
t=abs(diffrernce);
t0=sum(sum(t<=0.5));
t1=sum(sum(t<=1))-t0;
t2=sum(sum(t<=2))-t0-t1;
t3=sum(sum(t<=3))-t0-t1-t2;
t4=sum(sum(t<=5))-t0-t1-t2-t3;
t5=sum(sum(t>5));
out=0.5*t1+1.5*t2+6*t3+12*t4+25*t5;
end


function [out] =evaluate(in,Y)%计算成本
out=length(in)*12;
X=5:0.1:10;
xx=5+(in-1)*0.1;
yy=Y(:,in);
f=spline(xx,yy);
difference=ppval(f,X)-Y;
out=out+errorcost(difference)/469;
```

```
fid=fopen('result.txt','a');
fprintf(fid,'position: [ ');
fprintf(fid,'%2d ',in);
fprintf(fid,']        mean_cost: %7f\n\n',out);
fclose(fid);
end
```