统计推断在数模转换系统中的应用

组号：40
姓名：曹哲汉（组长） 5140309326
姓名：郑禹君（组员） 5140309363

摘要：本报告主要展示了统计推断在数模转换系统中的应用。以上海交通大学电院电子系实验所测得的共计 400 组输入信号 X 与输出 Y 的关系，运用一定的数理统计方法，经过特征点选取、算法研究、拟合比较等一系列过程，借助 Matlab 建立函数曲线模型，终实现以少量数据反映整体系统特性的效果，从而有效降低工程设计上的成本，提高效率。

关键词：样本，特征点，曲线拟合，样条插值拟合，遗传算法

The application of statistical reference in the system of DA conversion

ABSTRACT: This report is mainly about the application of statistical inference processing the data in DA transformation. By the method of mathematical statistics, with procedures of characteristic point selecting, Algorithm Researching and fitting comparing, relation curve model between the input and the output can be established by Matlab. Finally, we can find a way by which an entire system character will be reflected with less data measuring, thus effectively reduce the engineering design of the cost and improve efficiency.

Key words: sample, feature points, polynomial fitting, Spline difference fitting, genetic algorithm

1. 引言

1.1 课题概述

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设计一种成本合理的传感特性校准（定标工序）方案。

1.2 课程基本问题

针对某一特定传感部件个体，通过有限次测定，估计其检测的对象 Y 值与传感器输出电压信号 X 值间一一对应的特性关系 的过程。要求总成本（测定成本与误差成本的和）较低。

1.3 传感部件特性

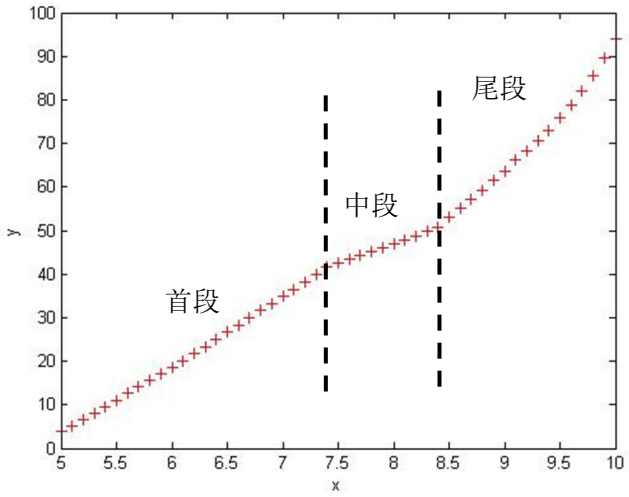


图 1-3 传感特性图示

一个传感部件个体的输入输出特性大致如图 2 所示，有以下主要特征：

- Y 取值随 X 取值的增大而单调递增；
- X 取值在[5.0,10.0]区间内，Y 取值在[0,100]区间内；
- 不同个体的特性曲线形态相似但两两相异；
- 特性曲线按斜率变化大致可以区分为首段、中段、尾段三部分，中段的平均斜率小于首段和尾段；
- 首段、中段、尾段单独都不是完全线性的，且不同个体的弯曲形态有随机性差异；
- 不同个体的中段起点位置、终点位置有随机性差异。

2. 评价标准的构建

2.1 成本计算

- 单点定标误差成本

$$s_{i,j}=\begin{cases}0 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|\le 0.4\\ 0.1 & \text{if } 0.4<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 0.6\\ 0.7 & \text{if } 0.6<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 0.8\\ 0.9 & \text{if } 0.8<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 1\\ 1.5 & \text{if } 1<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 2\\ 6 & \text{if } 2<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 3\\ 12 & \text{if } 3<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 5\\ 25 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|>5\end{cases} \quad (1)$$

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=12。

- 某一样本个体的定标成本

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

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。

- 校准方案总成本

按式（3）计算评估校准方案的总成本，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。

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

总成本较低的校准方案，认定为较优方案。

2.2 插值拟合分析

运用 matlab 实现插值拟合，伪代码为：

（1）产生一组数据中 n 个随机特征点。

（2）通过 matlab 函数 yi=(X, V, Xq, METHOD)获取全部数据的拟合值，首先先利用 spline，进行评价比较。

（3）处理所有数据，最后得出平均评价分数。

（4）改变 METHOD， 选用 spline， pchip(cubic)， nearest，分别拟合比较。

下给出插值拟合处理一组数据的图像。(取点为 7 个，蓝线代表原始数据连线，红线代表拟合之后连线)

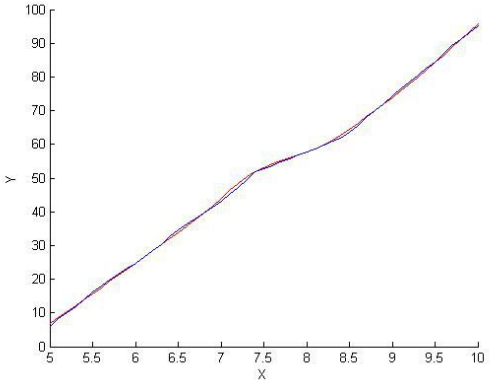


图 2-2-1 pchip 拟合方式图像

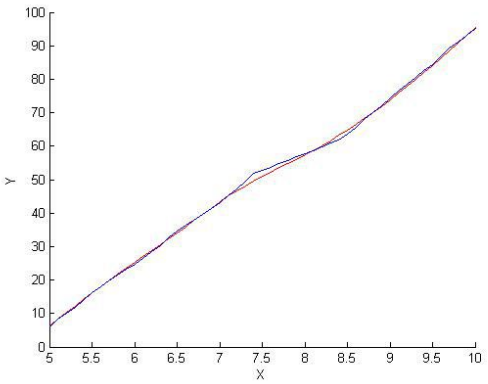


图 2-2-2 spline 拟合方式图像

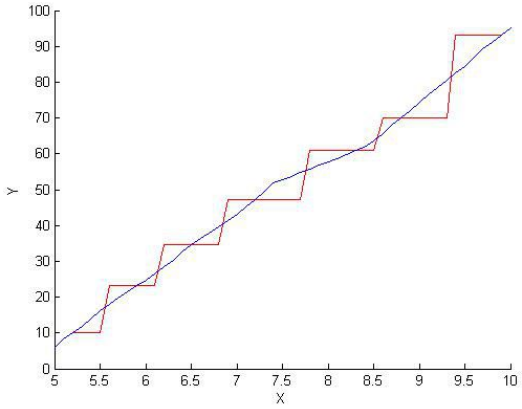


图 2-2-3 nearest 拟合方式图像

2.3 选取特征点

选取特征点想得到更为精确的拟合结果，需要选择更优的拟合方式。在实验中，我们采用插值拟合，我们直接使用 matlab 提供的拟合函数进行拟合。再将数据大致分为若干段，在每一段中，采取随机数的方法各选取一个数字作为特征点下标，使用此组下标作为所有样本数据拟合所用特征点下标。如果采用将所有可能性计算的话，数据量太大无法通过 PC 机实现。考虑到特征点分布不会太集中而是比较均匀，所以可以将 51 个数据平均分为 n 组(n=4,5,6,7,8)，在每组中选取特征点，这样大大减少了计算量。下以取 5 个点为例。

观测点序号	选取区间
1	[1, 8]
2	[9，19]
3	[20， 33]
4	[34， 43]
5	[44， 51]

表 2-3 观测点选取范围

3. 用遗传算法寻找最优组合

3.1 遗传算法简介

遗传算法（Genetic Algorithm）是模拟达尔文生物进化论的自然选择和遗传学机理的生物进化过程的计算模型，是一种通过模拟自然进化过程搜索最优解的方法。遗传算法是从代表问题可能潜在的解集的一个种群（population）开始的，而一个种群则由经过基因（gene）编码的一定数目的个体(individual)组成。每个个体实际上是染色体(chromosome)带有特征的实体。染色体作为遗传物质的主要载体，即多个基因的集合，其内部表现（即基因型）是某种基因组合，它决定了个体的形状的外部表现，如黑头发的特征是由染色体中控制这一特征的某种基因组合决定的。因此，在一开始需要实现从表现型到基因型的映射即编码工作。由于仿照基因编码的工作很复杂，我们往往进行简化，如二进制编码，初代种群产生之后，按照适者生存和优胜劣汰的原理，逐代（generation）演化产生出越来越好的近似解，在每一代，根据问题域中个体的适应度（fitness）大小选择（selection）个体，并借助于自然遗传学的遗传算子（genetic operators）进行组合交叉（crossover）和变异（mutation），产生

出代表新的解集的种群。这个过程将导致种群像自然进化一样的后生代种群比前代更加适应于环境，末代种群中的最优个体经过解码（decoding），可以作为问题近似最优解。

遗传算法的基本运算过程如下：

a)初始化：设置进化代数计数器 generation=0，设置最大进化代数 generationmax=20，设置种群总数 popsize=100，染色体数 chromo=n(n=4,5,6,7,8),变异概率 pm=0.005 以及交叉概率 px=0.8, 读入文件数据 data，初始化种群 initialize 以及总成本 totalcost=0。

b)个体评价：计算个体的总成本 singlecost。

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：将交叉算子作用于群体。遗传算法中起核心作用的就是交叉算子。

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。

f)终止条件判断:若 generation = generationmax, ,则以进化过程中所得到的具有最小成本个体作为最优解输出，终止计算。如不中止，则将变异之后的新群体带回到步骤（2），实现循环。

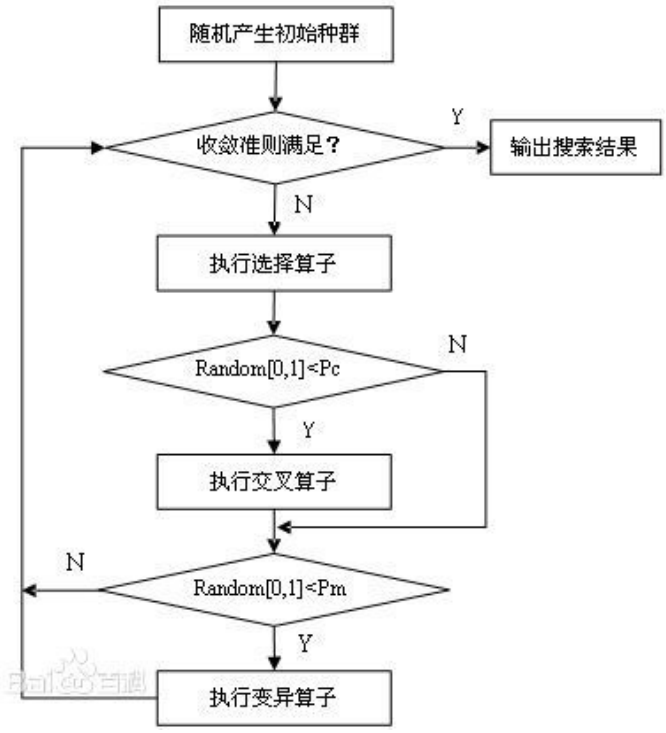


图 3-1 遗传算法流程图

4. 结论

表 4-1 遗传算法 20 组中最小的 5 组最优点

组号	成本	最优点					时间
1	84.75	4	16	26	35	48	717.878501s
2	84.75	4	16	26	35	48	221.082073s
3	84.75	4	16	26	35	48	324.841054s
4	84.75	4	16	26	35	48	315.905005s
5	84.75	4	16	26	35	48	156.97169s

由表 4-1 可以看出，本小组采取的分 5 段取点,pchip 插值和遗传算法充分利用了数据的性质，有效的解决了数据量大，耗时长的问题。同时只要遗传代数足够多，选取的亲本够多，则成本还可以更低。但是我们还应看到，在特征点之外的数据无法对结果产生影响，从而导致数据的遗漏。

5. 拓展一 特征点的个数选取

在 5 个特征点可以满足本课题要求的前提下，我们可以猜想更少的点应该也可以满足 要求，所以通过改变程序取点的个数我们做出了 4 个、6 个、7 个、8 个特征点的结果。4 个特征 点时选取区间重新分为[1,9]、[10,23]、[24,38]、[39,51]，6 个特征 点的选取区间为[1,6]、[7,12]、[13,25]、[26,38]、[39,44]、[45,51]，7 个特征点的选取区间为[1,8]、 [9,15]、[16,22]、[23,29]、[30,36]、[37,43]、[44,51],8 个特征点的选取区间为[1,6]、 [7,12]、[13,18]、[19,24]、[25,30]、[31,36]、[37,42]、[43,51]。将十次中最低成本得到下表所示结果。

取点数	成本	最优点								时间
4	100.43	4	21	34	47					147.670674s
6	88.01	3	15	22	29	39	50			155.675863s
7	91.97	3	13	21	27	34	43	50		165.268929s
8	101.31	2	12	18	24	29	35	42	50	164.798919s

表 5-1 选取不同特征点的结果

随着特征点数目的减少，寻求满足要求的特征点难度增大，并且成本逐渐变高。我们记录了表 5-1 中的不同特征点的最低成本以及对应的特征点。我们看出描述该问题的特征点可以选择 6 个点，且 6 个点的成本只是稍高于 5 个点的，所以特征点可以选取 5 个或 6 个。

6. 拓展二 用模拟退火法选取特征点

6.1 模拟退火法简介

模拟退火算法(Simulated Annealing，SA)最早的思想是由 N. Metropolis 等人于 1953 年提出。1983 年,S. Kirkpatrick 等成功地将退火思想引入到组合优化领域。它是基于 Monte-Carlo 迭代求解策略的一种随机寻优算法，其出发点是基于物理中固体物质的退火过程与一般组合优化问题之间的相似性。模拟退火算法从某一较高初温出发，伴随温度参数的不断下降,结合概率突跳特性在解空间中随机寻找目标函数的全局最优解，即在局部最优解能概率性地跳出并最终趋于全局最优。模拟退火算法是通过赋予搜索过程一种时变且最终趋于零的概率突跳性，从而可有效避免陷入局部极小并最终趋于全局最优的串行结构的优化算法。

6.2 模拟退火法的模型

1 模拟退火算法可以分解为解空间、目标函数和初始解三部分。

2 模拟退火的基本思想:

(1) 初始化：初始温度 T(300)，初始解状态 costmin(是算法迭代的起点)，每个 T 值的迭代次数 L

(2) 对 k=1，……，L 做第(3)至第 6 步：

(3) 产生新解 newcost

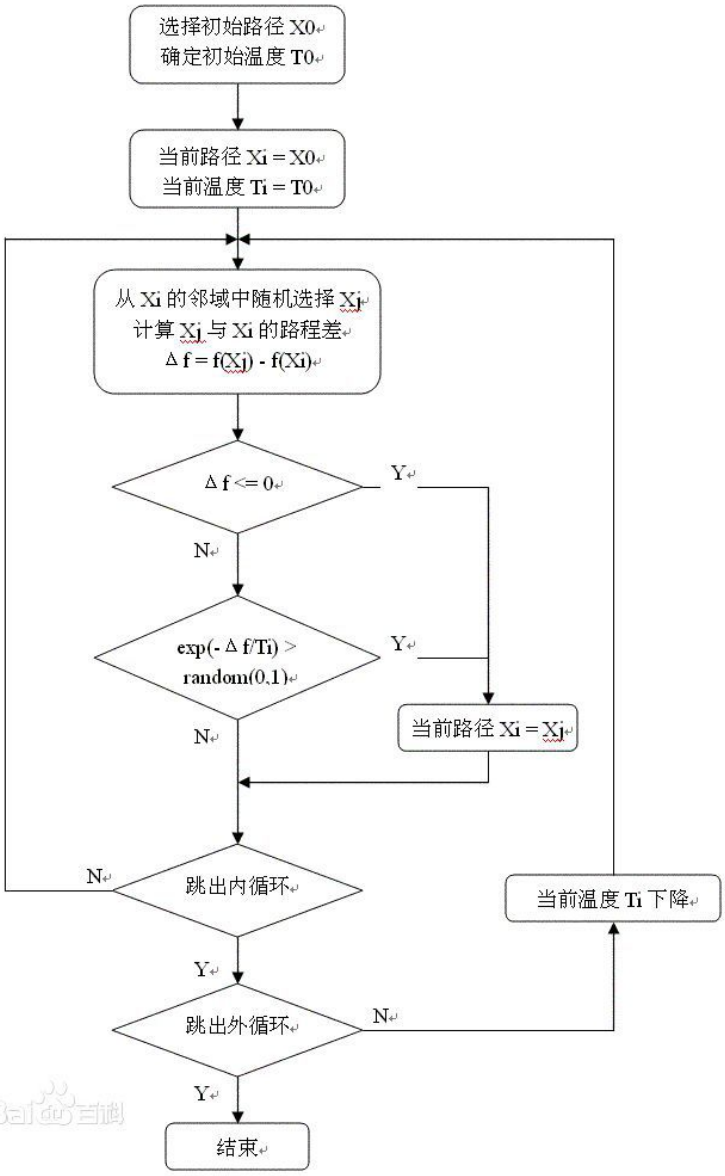
(4) 计算增量 d=newcost - costmin

(5) 若 d<0 则接受 S′作为新的当前解，否则以概率 exp(-d/T)接受 newcost 作为新的当前解.

(6) 如果满足终止条件则输出当前解作为最优解，结束程序。

终止条件通常取为连续若干个新解都没有被接受时终止算法。

(7) T 逐渐减少，且 T->0，然后转第 2 步



[

图 6-1 模拟退火算法流程图

6.3 结论

选取测试的 20 次中成本最低的 4 次记录出下表所得数据

表 6-3 模拟退火 20 次中成本最低 4 次取点方案及其成本

组号	成本	最优点					时间
1	85.15	4	15	25	35	48	26.249196s
2	85.02	5	16	26	35	48	26.249197s
3	85.14	4	16	26	35	49	26.249198s
4	84.75	4	16	26	35	48	26.249199s

由表 6-3 可以看出模拟退火算法和遗传算法的成本差不多，但是模拟退火算法的所需时间只有二十几秒秒，运行很快。

7. 鸣谢

非常感谢袁炎老师的指导以及其他小组的帮助，从刚接触 matlab 和统计推断方法的陌生到现在可以较熟练的应用 matlab 计算和作图离不开老师、助教和同学们的帮助。感谢在面谈的过程中，助教对我算法可以改进的部分以及终稿的悉心指导。同时，也感谢学姐流传下来的优秀报告，让我们可以尽快熟悉报告类型和模板，站在巨人的肩膀。

8. 参考文献

[1] 上海交通大学电子工程系统计推断讲座 ppt.
[2] 刘浩， 韩晶 《MATLAB R2014a 完全自学一本通》 电子工业出版社
[3] “统计推断”课程设计的要求 V2.2 2015-9-22.doc
[4] 陈国良.遗传算法及其应用.人民邮电出版社,1996.06
[5] 姚新.陈国良.模拟退火算法及其应用.中国科技大学计算机科学技术系.计算机研究与发展,1990.07
[6] 百度百科遗传算法词条：http://baike.baidu.com/view/45853.htm
[7] 百度百科模拟退火算法词条：http://baike.baidu.com/view/335371.htm
[8] 统计推断-第 39 组（李晔璇）课程设计报告

9. 附录

9.1 遗传算法代码：

```
clc
clear all
close all
tic
data = csvread('20150915dataform.csv');
chromo = 5;
generationmax = 20;
px = 0.8;%交叉概率
pm = 0.005;%变异概率

pop_size = 200;
generation = 1;
totalcost = zeros(1, pop_size);
population = initialize(pop_size, chromo);

for i = 1 : 1 : pop_size
    totalcost(i) =  mycost(population(i,:), data);
end
```

```
mincost = totalcost(pop_size);

while generation <= generationmax
    [population, totalcost] = mutate(population, totalcost, pop_size, pm,
data, chromo);%变异
    [totalcost, Index] = sort(totalcost, 'descend');
    population = population(Index, :);

    [population, totalcost] = crossover(population, totalcost, pop_size,
px, chromo, data);%交叉
    [totalcost, Index] = sort(totalcost, 'descend');
    population = population(Index, :);
    [population, totalcost] = select(population, totalcost, data,
chromo);%选择
    if mincost > totalcost(pop_size)
        mincost = totalcost(pop_size);
        P = population(pop_size, :);
    end

    disp('generation :');
    disp(generation);
    disp('对应成本为')
    disp(mincost);
    disp('取点方案为')
    disp(P);
    generation = generation + 1;
end
toc
function [population] = initialize(pop_size, chromo)
population = zeros(pop_size, chromo);
switch chromo
    case 8
        population = zeros(pop_size, chromo);
        for i=1:1:pop_size
            population(i, 1) = ceil(rand*6);
            population(i, 2) = 6 + ceil(rand*6);
            population(i, 3) = 12 + ceil(rand*6);
            population(i, 4) = 18 + ceil(rand*6);
            population(i, 5) = 24 + ceil(rand*6);
            population(i, 6) = 30 + ceil(rand*6);
            population(i, 7) = 36 + ceil(rand*6);
            population(i, 8) = 42 + ceil(rand*8);
        end
    case 7
```

```
        population = zeros(pop_size, chromo);
        for i=1:1:pop_size
            population(i, 1) = 1 + ceil(rand*7);
            population(i, 2) = 8 + ceil(rand*7);
            population(i, 3) = 15 + ceil(rand*7);
            population(i, 4) = 22 + ceil(rand*7);
            population(i, 5) = 29 + ceil(rand*7);
            population(i, 6) = 36 + ceil(rand*7);
            population(i, 7) = 43 + ceil(rand*7);
        end
    case 6
            population = zeros(pop_size, chromo);
        for i=1:1:pop_size
            population(i, 1) = ceil(rand*6);
            population(i, 2) = 6 + ceil(rand*6);
            population(i, 3) = 12 + ceil(rand*13);
            population(i, 4) = 25 + ceil(rand*13);
            population(i, 5) = 38 + ceil(rand*6);
            population(i, 6) = 44 + ceil(rand*7);
        end
    case 5
        population = zeros(pop_size, chromo);
        for i=1:1:pop_size
            population(i, 1) = ceil(rand*8);
            population(i, 2) = 8 + ceil(rand*11);
            population(i, 3) = 19 + ceil(rand*14);
            population(i, 4) = 33 + ceil(rand*10);
            population(i, 5) = 43 + ceil(rand*8);
        end
    case 4
        population = zeros(pop_size, chromo);
        for i=1:1:pop_size
            population(i, 1) = ceil(rand*10);
            population(i, 2) = 10 + ceil(rand*13);
            population(i, 3) = 23 + ceil(rand*15);
            population(i, 4) = 38 + ceil(rand*13);
        end
end
end
function [cost] = mycost(P, minput)
sort(P);
my_answer=P;%取点方案
my_answer_n=size(my_answer,2);
[M,N]=size(minput);
```

```
nsample=M/2; npoint=N;
x=zeros(nsample,npoint);
y0=zeros(nsample,npoint);
y1=zeros(nsample,npoint);
for i=1:nsample
    x(i,:)=minput(2*i-1,:);
    y0(i,:)=minput(2*i,:);
end
my_answer_gene=zeros(1,npoint);
my_answer_gene(my_answer)=1;


% 定标计算
index_temp=logical(my_answer_gene);
x_optimal=x(:,index_temp);
y0_optimal=y0(:,index_temp);
for j=1:nsample
    y1(j,:)=mycurvefitting(x_optimal(j,:),y0_optimal(j,:));
end
 %成本计算
Q=12;
errabs=abs(y0-y1);

le0_4=(errabs<=0.4);
le0_6=(errabs<=0.6);
le0_8=(errabs<=0.8);
le1_0=(errabs<=1);
le2_0=(errabs<=2);
le3_0=(errabs<=3);
le5_0=(errabs<=5);
g5_0=(errabs>5);

sij=0.1*(le0_6-le0_4)+0.7*(le0_8-le0_6)+0.9*(le1_0-le0_8)+1.5*(le2_0-
le1_0)+6*(le3_0-le2_0)+12*(le5_0-le3_0)+25*g5_0;
si=sum(sij,2)+Q*ones(nsample,1)*my_answer_n;
cost=sum(si)/nsample;
function y1 = mycurvefitting( x_premea,y0_premea )

x=[5.0:0.1:10.0];

% 将你的定标计算方法写成指令代码，以下样式仅供参考
y1=interp1(x_premea,y0_premea,x,'pchip');
```

```
end
function [population, totalcost] = crossover(population, totalcost,
pop_size, px, chromo, data)
for i = pop_size : -2 : (pop_size/2+2) %优秀个体交叉
    sort(population(i, :)); %重排顺序
    sort(population(i-1, :));
    if rand < px
        point = randi(chromo);  %单点交叉
        child1 = [population(i, 1:point), population(i-1,
point+1:chromo)]; %组合
        child2 = [population(i-1, 1:point), population(i,
point+1:chromo)];
        child1 = cherk(child1, chromo);
        child2 = cherk(child2, chromo);
        if
(length(unique(child1))==chromo)&&(length(unique(child2))==chromo) %没
有重复元素否则交叉失败 population 还有 totalcost 不变
            cost1 = mycost(child1, data);
            cost2 = mycost(child2, data);
            if totalcost(i) > cost1 %第 i 个个体的替换
                totalcost(i) = cost1;
                population(i,:) = child1;
            end
            if totalcost(i-1) > cost2  %第 i+1 个个体的替换
                totalcost(i-1) = cost2;
                population(i-1,:) = child2;
            end
            sort(population(i, :)); %打乱顺序后重排
            sort(population(i-1, :));
        end
    end
end

for i = pop_size/2 : -1 : 1 %适应度低的个体交叉
    sort(population(i, :)); %重排顺序
    if rand < px
        point = ceil(rand*4);  %单点交叉
        child1 = [population(i, 1:point), population(i+pop_size/2,
point+1:chromo)]; %组合
        child1 = cherk(child1, chromo);
        cost1 = mycost(child1, data);
        if totalcost(i) > cost1 %第 i 个个体的替换
                totalcost(i) = cost1;
                population(i,:) = child1;
```

```
        end
        sort(population(i, :)); %打乱顺序后重排
    end
end
end
function  [population, totalcost] = mutate(population, totalcost,
pop_size, pm, data, chromo)
for i = 1 : 1 : pop_size
    if rand < pm
        while 1
            if totalcost(i) < (chromo*12 + 10)
                len = randi([-2, 2]);
            else
                len = randi([-5, 5]);
            end
            pos = randi(chromo);
            child = population(i,:);
            child(1, pos) = mod(population(i, pos)+len, 50)+1;
            if length(unique(child)) == chromo
                cost1 = mycost(child, data);
                if totalcost(i) > cost1
                    totalcost(i) = cost1;
                    population(i, :) = child;
                end
                break;
            end
        end
    end
end
end
function [population, totalcost] = select(population, totalcost, data,
chromo)
population([1,2,3,4],:) = initialize(4, chromo);
totalcost(1) = cost(population(1,:), data, chromo);
totalcost(2) = cost(population(2,:), data, chromo);
totalcost(3) = cost(population(3,:), data, chromo);
totalcost(4) = cost(population(4,:), data, chromo);
end
function[x] = cherk(x, chromo)
sort(x);
new = unique(x);
if length(new) ~= chromo
    x = initialize(1, chromo);
end
```

```
end
```

9.2 模拟退火算法代码：

```
T = 300;
n = 5;
rate = 0.97;
Tout = 5;
data = csvread('20150915dataform.csv');
L = 0;
X = initialize(1, n);
costmin = mycost(X, data);
tic
while (T>Tout)
    Xnew = X + round((rand(1, n)-0.5)*2);
    while
(Xnew(1)<1||Xnew(n)>51||abs(Xnew(1)-Xnew(2))<3||abs(Xnew(2)-Xnew(3))<
3||abs(Xnew(2)-Xnew(3))<3||abs(Xnew(3)-Xnew(4))<3||abs(Xnew(4)-Xnew(5)
)<3)
        Xnew = X + round((rand(1, n)-0.5)*2);
    end
    newcost = mycost(Xnew, data);
    d = newcost - costmin;
    if (d<0)
        X = Xnew;
        costmin = newcost;
    else
        p = exp((-d)*1000/T);
        if (rand < p)
            X = Xnew;
            costmin = newcost;
        end
    T = T*rate;
    end
    L = L + 1;
    disp(costmin);
end
toc
disp(X);
disp(costmin);
```

其余初始化与成本计算函数与遗传算法相同