# 统计推断在数模转换系统中的应用

第 24 小组 赵璟浩 5140309102 何昊锟 5140309117

**摘要：**统计推断在数模模数中有重要应用。本课题围绕传感器部件的特性校准方案展开，我们运用遗传算法以及退火算法对定标点进行合适的选取，再通过尝试不同拟合方式，寻求最小成本，运用 matlab 进行编程实现这一思想。

**关键词：**统计推断，遗传算法，退火算法，定标，拟合，matlab

## 参考他人报告或代码的申明：

统计推断课程，2015 年秋季学期第 24 组，成员赵璟浩学号 5140309102, 何昊锟学号 5140309117, 在报告编写过程中，以下方面参考了往届报告，现列表说明：

| 主要参考项目 | 说明 |
|---|---|
| 代码方面 | 参考原文：《统计推断在数模模数转换中的应用》，黄博天，2014 年秋季学期，组号 45<br>编写遗传算法代码时，在其附录中程序代码基础上做了优化，重新编写适应度函数。 |

# 1 引言

本课题要求为一电子产品的监测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。

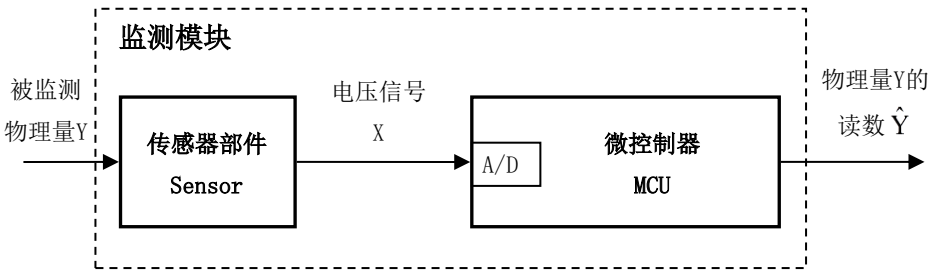


图 1 监测模块模型[1]

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 $y_i$， $i = 1,2,3,...,50,51$

在本组课题中我们采用 excel 表格中的 400 组，每组 51 个数据作为监测模块的数据样本。

# 2 实现步骤

## 2.1 选择插值方法

在我组课题中我们采用 matlab 中的三次插值函数进行插值，调用格式为：

yi= interp1(x,y,xi,'method')

其中 x，y 为插值点，yi 为在被插值点 xi 处的插值结果；x,y 为向量，“method”表示采用的插值方法，本课题通过采用“spline”——三次样条插值以及“cubic”——三次插值类型，拟合计算后来取得最小成本。

需要注意的是，所有的插值方法都要求 x 是单调的，并且 xi 不能够超过 x 的范围。因此，在使用多次样条差值时，取样测定点应包含首尾两点。

## 2.2 选择取点方式

模拟退火算法中我们采取这样的取点方式。

由于每组数据有 51 个样本点，考虑到在实际测量工作中采样的工作量以及成本，故每组数据选取一些点来代表 51 个点。在我组课题中，根据传感部件的特性，传感部件的特性曲线图可分为三个部分，前、中、后段。其中，中段的斜率低于前段与后段，比较平缓，故我们的取点分为三段：[5.0,5.9]，[6.0,9.0]，[9.1,10.0]，在这三段中，在第二段选取较多一点数据点，第一与第三段选取较少一点数据点。下表为具体选取方式之一：

| 区段 | X 范围 | 对应采样序号点 | 采样数据点数目 |
|---|---|---|---|
| 1 | [5.0,5.9] | 1,2,3,…,9 | 2 |
| 2 | [6.0,9.0] | 10,11,12,…,41 | 3 |
| 3 | [9.1,10.0] | 42,43,44,…,51 | 2 |

我们最终选取了上述取点方式作为退火算法的初始点。

## 2.3 定标

### 2.3.1 选择拟合方式

首先我们在不进行插值的情况下，对取到的数据点直接进行拟合，拟合方式为高次多项式拟合，利用 matlab 的 polyfit 函数可以对数据进行高次多项式拟合，其中 polyfit 函数调用的形式为 polyfit(x,y,m)。表示的是用 m 次多项式来拟合数据 x 和 y。首先我们对已采取的数据点进行高次拟合，此次课题中我们采用插值的方法，利用 matlab 中的一位数据的插值函数：interp1( ) 可以实现插值拟合。该函数的调用方式为 y1=interp1(x,y,x1,methods)。其中 x,y 分别表示一组数据点的横纵坐标， x1 为插入的横坐标， methods 为可选参数，再对取到的数据点，采用“spline”——三次样条插值以及“cubic”——三次插值，再进行上述的拟合。

### 2.3.2 具体定标

根据拟合的函数，计算出 x = [5.0,10.0] 中的 51 个函数值，与样本实际数据值计算差值。

## 2.4 计算成本[2]

### 2.4.1 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。

### 2.4.2 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。

### 2.4.3 某一样本个体的定标成本

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

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。

### 2.4.4 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总成本较低的校准方案，认定为较优方案。

$$C = \frac{1}{M}\sum_{i=1}^{M} S_i \tag{3}$$

# 3 算法

## 3.1 退火算法

### 3.1.1 算法概述

模拟退火算法模拟金属退火时的过程，即将金属加热到一定温度，再经过特定速率的冷却，这样在缓慢冷却的过程中原子会找到比之前状态内能更低的位置。算法在搜寻的过程中先以任意点作为起始，每一步先选择一个临近的数据点，然后再计算从现有位置到临近位置的概率。在温度较高时，系统有较大概率接受误差较大的分子，这样有利于避免局部收敛的情况，随着温度的降低，数据点逐渐趋于稳定。可以证明，模拟退火算法所得结果依概率收敛到全局最优解。

### 3.1.2 算法设计过程

退火算法的基本运算过程如下：

a)初始化：随机产生一数组作为初始值，选择时做一简单的筛选，使得初始成本降低到可以接受的范围内，减少程序工作量。

b)设定初始值：设定初始温度为 100T，每次循环温度降低为原温度的 0.98，直到温度降低至 0.00001T 时终止

c)迭代过程：

1.每次迭代中先调整第一个元素的位置，然后调整中间五个元素的位置，最后调整后一个元素的位置，调整时只对元素进行相邻的调整，即只将选点位置变换为其原位置的相邻位置。

2.若新产生的数组成本小于原最优数组，则最优数组变为新产生数组。

3.若成本高于最优数组，利用退火突变概率公式e = exp(-abs(bestScore-score) / T)，求得当前突变概率，利用随机函数决定是否突变。

5.输出当前最优点。

d)输出最优点方案。

### 3.1.3 退火算法的代码实现

退火算法由 matlab 上编程进行算法的实现。实现程序中包含了主函数 SAAmain 和计算成本的函数 get_score，具体代码实现附后。

## 3.2 遗传算法

### 3.2.1 算法过程概述

遗传算法的基本运算过程如下：

**a)初始化：**在本课题中，首先需要解决的是，如何将取点方案数字化并且能够尽量简单方便可行。我们将每一个取点方案称为一个个体，参照遗传学机制的生理进化过程，每个个体实际上是染色体带有特征的实体，每个染色体上带有一定数量的基因，我们现在需要解决的即如何初始化这些基因。 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。

在本课题中，由于样本数据库中的每组数据有 51 个数据点，我们将每个染色体个体的二进制字符串的长度设置为 51，每一位数字代表当前位点的取点是与否。0 代表取点时不包括该点，1 表示取点时包括该点。将第 1 位与第 51 位数字设置成 1，中间的二进制序列随机生成为 0 或 1。

设置进化代数计数器 t=0，设置最大进化代数 T=100 代，随机生成 30 个个体作为初始群体 P(0)。

**b)个体评价：**在每一代中，如 t 代时，计算群体 P(t)中各个个体的适应度。由于本课题需要设计出一个最终成本最低的取点方案，故我们采用成本分之一（即倒数）作为每个个体的适应度。

**c)选择运算:**在有了随机产生出的染色体个体组后，需要设置一个合理的选择方式，从而实现优胜劣汰的进化过程。根据达尔文进化论，越能适应环境的个体就越有可能繁衍。我们选用最简单最常用的“轮盘赌选择法”。设个体适应度为 fi，则其被选中的概率 Pi 满足以下公式：

$$P_i = f_i / \sum_{j=1}^{n} f_j$$

**d)交叉运算：**在自然界生物进化过程中起核心作用的是生物遗传基因的重组(以及变异)。同样，遗传算法中起核心作用的是遗传操作的交叉算子。所谓交叉是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。我组课题中的二进制编码的基因交换过程也借鉴了这个过程－－即随机选取两条染色体，在染色体第 2 至第 50 个位置中随机选取一个点作为交叉点，将两个染色体分为四段，，将交叉点后的部分互相交换，产生新的两个染色体个体。本组将杂交概率设置为 0.9，即两两染色体之间有 90%的概率发生上述过程。

**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。参照遗传学机制的变异机制，即基因中碱基改变，我组课题中二进制编码的变异操作设定为变异位数的二进制编码的反码，即将“0”转变为“1”，“1”转变为“0”。程序中变异概率设置为 0.05。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)。

**f)终止条件判断:**若 t=100,到达设置的进化代数，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。下图是遗传算法的流程示意图：

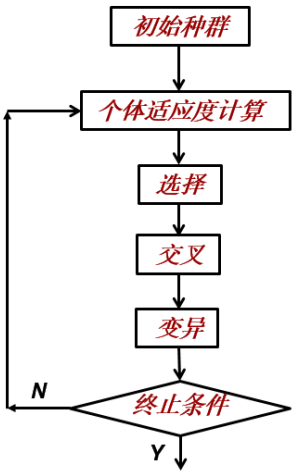


图 2 遗传算法流程图[3]

### 3.2.2 遗传算法的代码实现

在设计出算法之后，我们在 matlab 上编程进行算法的实现。实现程序中包含了拟合、计算成本过程 get_score，基因交叉过程 crossfun，基因突变过程 mutationfun，以及整体循环 GAmain。

其中需要说明的是：get_score 选取了三次样条插值法进行计算；crossfun 采用了 randperm 函数对所有取点方案进行随机编号，再随机选取交叉点；mutationfun 设置了一个很小的突变概率 pm，再随机选取一个点进行突变,；主函数中包含了初始化过程与循环中的选择过程，初始时，除了首尾两点外，我们随机设置其中点为 1。循环过程中，每次记录下本次循环中最佳取点方案与 30 组取点方案的平均成本，输出后便于比较。适应度即为 fitnessfun 函数的返回值的倒数。

# 4 结果分析

## 4.1 运行结果

### 4.1.1 退火算法 cubic 拟合

| 取点方案 | 分数 | 时间 |
|---|---|---|
| 3， 13， 21， 27， 34， 43， 50 | 91.9680 | 72.66s |
| 3， 13， 21， 27， 34， 43， 50 | 91.9680 | 71.83s |
| 4， 12， 21， 27， 34， 42， 51 | 92.6575 | 47.81s |
| 3， 13， 21， 27， 34， 43， 50 | 91.9680 | 71.36s |
| 3， 13， 21， 27， 34， 43， 50 | 91.9680 | 71.93s |

表 1 退火算法 cubic 拟合结果

### 4.1.2 退火算法 spline 拟合

| 取点方案 | 分数 | 时间 |
|---|---|---|
| 2， 9， 20， 27， 34， 44， 50 | 95.2147 | 87.20s |
| 2， 9， 19， 26， 33， 43， 50 | 95.1147 | 75.84s |
| 2， 9， 19， 26， 33， 43， 50 | 95.1147 | 74.50s |
| 2， 9， 19， 26， 33， 43， 50 | 95.1147 | 93.67s |
| 2， 9， 19， 26， 33， 43， 50 | 95.1147 | 78.07s |

表 2 退火算法 spline 拟合结果

### 4.1.3 遗传算法 cubic 拟合

| 取点方案 | 分数 | 时间 |
|---|---|---|
| 1， 15， 27， 34， 44， 51 | 91.9680 | 283.97s |
| 1， 16， 24， 33， 43， 51 | 91.0660 | 280.51s |
| 1， 13， 23， 30， 40， 51 | 91.6312 | 243.58s |
| 1， 13， 23， 30， 37， 51 | 91.4117 | 252.26s |
| 1， 14， 23， 29， 39， 51 | 90.2315 | 241.98s |

表 3 遗传算法 cubic 拟合结果

### 4.1.4 遗传算法 spline 拟合

| 取点方案 | 分数 | 时间 |
|---|---|---|
| 1， 8， 19， 24， 31， 43， 51 | 98.9315 | 248.08s |
| 1， 4， 16， 25， 32， 45， 51 | 101.1163 | 255.74s |
| 1， 10， 22， 28， 35， 44， 51 | 97.9005 | 249.32s |
| 1， 9， 23， 26， 34， 44， 51 | 100.1650 | 254.84s |
| 1， 7， 17， 26， 30， 44， 51 | 100.7810 | 240.89s |

表 4 遗传算法 spline 拟合结果

## 4.2 结果对比

对两种拟合方法来说,每种算法分别运行了 5 次。由运行结果示例可以清晰地看出来，使用三次插值拟合法相比于三次样条插值拟合法有两点优势。1、使用前者，花费要低于后者，后者花费在 95 左右，而三次插值可以将成本控制在 90 左右。2、使用前者 5 次运行所花费的时间少于使用后者，因为程序的其他部分基本上都是一样的，原因可能是 matlab 中两种拟合函数内部运行时间不同。

两种算法对比来看，退火算法所用的时间要明显小于遗传算法，原因应该为内部具体循环时每个循环之中所用的步骤较少，也较为简单，没有过多的函数调用。成本方面两种算法均能较快收敛到最终的最优解，退火算法相比遗传算法中间步骤不确定性相对较小，故最终的结果较为稳定，所得最优解较为单一。

最终得到的最优选点方法为：1,14,23,29,39,51，在三次插值拟合下成本均值为90.23。

# 5 课程总结

在进行本课程实验前，我们组成员对于 matlab 以及对于具体数学模型的问题求解基本不了解，在参考前人的经验以及我们自己不断思考之后，我们逐渐地对于 matlab 编程以及问题的求解过程以及方式有了自己的认识与理解。在求解的过程中，我们遇到了许多问题，主要就是对于具体算法的理解与如何实现，即如何将算法的思想转化为实际的过程步骤以及在 matlab 上对算法进行代码实现。带着这两个问题，我们在参考前人的经验中得到自己的理解，之后不断的自己编写并调试熟悉代码，最终得以解决。在我看来，我们在本课题中收获的，并不仅仅是对于这一实际问题的具体求解方式。在做课题时，思考问题的求解步骤并实现，这一过程中，我们收获颇丰，我们对于 matlab 这一编程软件熟悉了许多，退火算法与遗传算法也对我们有了很多启发。当然，我们最终得到的结果也是令人满意的。以后相信遇到另一问题时，我们也能成功运用我们自己的智慧去解决。

# 6 参考文献

[1] “统计推断”课程设计的要求 V2.2 2015-9-22
[2] “统计推断”课程设计的要求 V2.2 2015-9-22
[3] 统计推断讲座 2_问题的提出和基本求解思路

# 7 附录

**退火算法代码：**

**主函数 SAAmain**

```
tic
clear all
close all

bestArray = zeros (1,7);% 最优解数组选点方案
bestScore = 0.0;
T = 100.0;%设定初始温度
score =150;
while score >=135 %找到一个成本小于135的点集，减少程序初始运行成本，并将其作为初始解
    arr1 = randi([1,5]);
    arr2 = randi([6,9]);
    arr3 = randi([10,20]);
    arr4 = randi([21,30]);
    arr5 = randi([31,41]);
    arr6 = randi([42,45]);
    arr7 = randi([46,51]);
```

```
    array = [arr1,arr2,arr3,arr4,arr5,arr6,arr7];
    score = get_score(array);
    bestScore = score;
    bestArray = array;
end
%模拟退货迭代
while T > 0.00001
    T = 0.95 * T;

    disp(score);
    %调整第1个元素的值
    while true
        cur = array(1);
        p = randi(5);
        if p == 5
           cur = cur + 1;
        elseif p == 1
            cur = cur - 1;
        end
        if cur >= 1 && cur < array(2)%限制变动范围在合法范围
            break
        end
    end
    array(1) = cur;
    %调整第2-6个元素的值
    for i = 2:6
        while true
            cur = array(i);
            p = randi(5);
            if p == 5
                cur = cur + 1;
            elseif p == 1
                cur = cur - 1;
            end
            if cur  > array(i - 1) && cur < array(i + 1)% 限制变动范围在合法范围
                 break
            end
        end
        array(i) = cur;
    end
    %调整第7个元素的值
    while true
        cur = array(7);
        p = randi(5);
```

```
        if p == 5
            cur = cur + 1;
        elseif p == 1
            cur = cur - 1;
        end
        if cur > array(6) && cur <= 51%限制变动范围在合法范围
            break
        end
    end
    array(7) = cur;
    disp(array);

    score = get_score(array);
    e = exp(-abs(bestScore-score) / T);

    if score <= bestScore
        bestScore = score;
        bestArray = array;

    elseif  rand>e %如果0-1之间随机数大于概率e，则数组保持不变
            array=bestArray;
    end
end
disp(bestScore);
disp(bestArray);
toc
```

**成本函数 get_score()**

```
function score=get_score(x)
score=0;
dataform = importdata('20150915dataform.csv');
i=2:2:800;
k=5:0.1:10;
data=dataform(i,:);
base = 5:0.1:10;

for index=1:400
    fit=interp1(k(x)',data(index,x)',base,'spline');
    sum=0;
    for j=1:51
        dif=abs(data(index,j)-fit(j));
        if dif<=0.4
            sum=sum+0;
        elseif dif<=0.6
```

```
            sum=sum+0.1;
        elseif dif<=0.8
            sum=sum+0.7;
        elseif dif<=1
            sum=sum+0.9;
        elseif dif<=2
            sum=sum+1.5;
        elseif dif<=3
            sum=sum+6;
        elseif dif<=5
            sum=sum+12;
        else sum=sum+25;
        end
    end
    score=score+sum+12*length(x);
end
score=score/400;
end
```

**遗传算法代码：**

**主函数 GAmain**

```
tic;
loopnum=100; %代数
number=30;  %个体数
pc=0.9;     %杂交概率
pm=0.05;    %变异概率
pop=zeros(number,51);  %繁殖空间
temp=zeros(number,51);
fitness=zeros(1,number);  %适应度
for i=1:number       %初始化
    for j=2:50
        if rand<0.1
            pop(i,j)=1;
        end
        pop(i,1)=1;
        pop(i,51)=1;
    end
end
for q=1:loopnum
    ave=0;    %平均值
    for i=1:number
        if length(find(pop(i,:)))<=2   %点数小于2时无效
            fitness(i)=0;
        else t=get_score(find(pop(i,:)));
```

```
            fitness(i)=1000/t;      %适应度
            ave=ave+t;
        end
    end
    ave=ave/number;
    [~,s]=max(fitness);      %最优个体
    fitness=fitness/sum(fitness);  %轮盘赌
    best=pop(s,:);
    fff=cumsum(fitness);
    for i=1:number
        t=find(rand<=fff);
        temp(i,:)=pop(t(1),:);
    end
    temp(number,:)=best;
    pop=temp;
    pop=crossfun(pc,pop);      %交叉
    pop=mutationfun(pm,pop);  %变异
    best_=zeros(1,1);
    j=1;
    for i=1:51
        if best(i)==1
            best_(j)=i;
            j=j+1;
        end
    end
    fprintf('代数:%d\n',q);
    fprintf('最优个体: %d\n');
    disp(best_);
    fprintf('最佳成本:%d\n');
    disp(get_score(find(best)));
    fprintf('平均成本:%d\n');
    disp(ave);
end
toc;
```

**交叉函数crossfun()**

```
 function [ newpop ] = crossfun(pc,pop)  %%个体之间进行交叉。
[px,py]=size(pop);
newpop=zeros(px,py);
a=randperm(px);
for i=1:2:(px-1)
    x=a(i);y=a(i+1);
    if(rand<pc)  %%以 pc 概率交叉
        cpoint=ceil(rand*49);  %随机交叉点
```

```
        newpop(x,:)=[pop(x,1:cpoint),pop(y,cpoint+1:py)];
        newpop(y,:)=[pop(y,1:cpoint),pop(x,cpoint+1:py)];
    else
        newpop(x,:)=pop(x,:);
        newpop(y,:)=pop(y,:);
    end
end
end
```

**变异函数mutationfun()**

```
function [ newpop ]= mutationfun( pm,pop )
px=size(pop,1);
newpop=zeros(size(pop));
for i=1:px
    newpop(i,:)=pop(i,:);
    if(rand<pm)
        mpoint=round(rand*50+1);     %突变点
        newpop(i,mpoint)=1-pop(i,mpoint);
    end
end
end
```

注：遗传算法计算适应度时，可以使用退火算法中get_score的计算成本函数，然后取倒数获得

**测试函数test_ur_answer**

```
%%%%%%% 答案检验程序 2015-11-04 %%%%%%%%

my_answer=[ 1,14,23,29,39,51 ];%把你的选点组合填写在此
my_answer_n=size(my_answer,2);

% 标准样本原始数据读入
minput=dlmread('20150915dataform.csv');
[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
    % 请把你的定标计算方法写入函数 mycurvefitting
    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;

% 显示结果
fprintf('\n 经计算，你的答案对应的总体成本为%5.2f\n',cost);
```

**拟合函数mycurvefitting()**

```
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,'cubic');

end
```