统计推断在数模转换系统中的应用

组号　34　　陈沐春 5130309284，孟文韬 5130309289，…

摘要：本文是上海交通大学电子信息与电气工程学院课程设计《统计推断在数模、模数转换系统中的应用》 的课程论文初稿。我们运用 Matlab 数学软件并使用统计推断的方法，通过 469 组数据对系统的输入输出关系进行研究，并使用 matlab 里的遗传算法工具箱对数据进行处理，找出最优解。

关键词：统计推断，MATLAB，遗传算法

Application of Statistical Inference in DA Inverting System

ABSTRACT: **This article is for Course Design-Application of Statistical Inference in AD&DA Inverting System, School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University. We use statistical inference, in combination with Matlab, to research into the relationship between the input and output , based on the given data. We manage to find fitting methods focusing on pre -observed points, in which genetic algorithm is applied.**

Key words: statistical inference, MATLAB,Genetic Algorithm

1 引言

本课题是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，为X和Y间的关系进行校准定标。

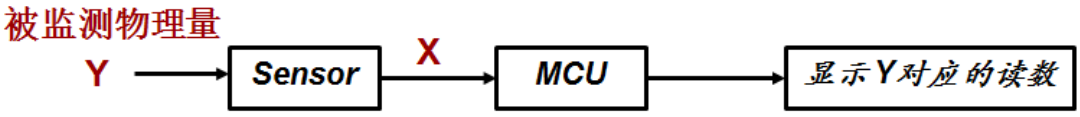

图 1-1　课题研究对象

我们把X为某个特定值时，针对相应Y的数值进行的测量过程，称作一次测定。而测定需要一定的成本。

根据几个样本的拟合图像，我们可以看出，Sensor特性呈现出非线性，而且个体与个体之间的差异还是比较大的，但是输入输出关系曲线总的变化趋势一定程度上是一致的（图 1-2），因此输入输出之间存在着某种非确定的关系。

通过观察可以发现曲线具有单调性，X取值都在5-10之间，Y取值都在0-100之间，曲线的斜率大小都满足先减小后增加即中断斜率小于两端的斜率，可以选取同样具有这种形状的曲线进行拟合。

由于实验所得的各组数据之间存在较大的差异，这增大了我们用统一的曲线表示X-Y关系的难度。在这篇课程论文中，我们希望借助于 Matlab 数学软件对本课题中的问题进行研究。

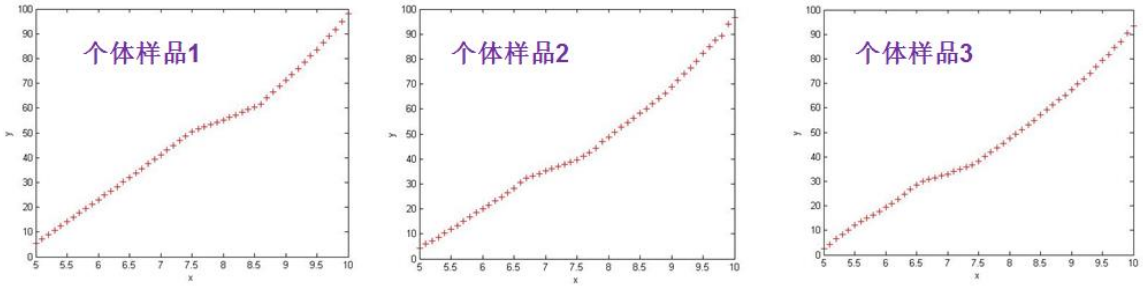

图 1-2　样品曲线

2 三次样条插值拟合

2.1 概述

早期工程师制图时，把富有弹性的细长木条（所谓样条）用压铁固定在样点上，在其他地方让它自由弯曲，然后沿木条画下曲线。成为样条曲线

三次样条插值（简称 Spline 插值）是通过一系列形值点的一条光滑曲线，数学上通过求解三弯矩方程组得出曲线函数组的过程。

2.2 相关函数

三次样条函数:

定义:函数 S(x)∈C2[a,b] ，且在每个小区间[xj,xj+1]上是三次多项式，其中

a =x0 <x1<...< xn= b 是给定节点，则称 S(x)是节点 x0,x1,...xn 上的三次样条函数。

若在节点 x j 上给定函数值 Yj= f (Xj).(j =0, 1, , n) ，并成立

S(xj) =yj .(j= 0, 1, , n) ，则称 S(x)为三次样条插值函数。

实际计算时还需要引入边界条件才能完成计算。边界通常有自然边界（边界点的二阶导为 0），夹持边界（边界点导数给定），非扭结边界（使两端点的三阶导与这两端点的邻近点的三阶导相等）。一般的计算方法书上都没有说明非扭结边界的定义，但数值计算软件如 Matlab 都把非扭结边界条件作为默认的边界条件。

2.3 Matlab 实现

interp1——一维数据插值函数

一维数据插值。该函数对数据点之间计算内插值，它找出一元函数 f(x)在中间点的数值，其中函数表达式由所给数据决定。

yi=interp1(x,Y,xi)：返回插值向量 yi，每一元素对应于参量 xi，同时由向量 X 与 Y 的内插值决定。参量 x 指定数据 Y 的点。若 Y 为一矩阵，则按 Y 的每列计算。yi 是阶数为 length(xi)*size(Y,2)的输出矩阵。

yi=interp1(Y,xi)：假定 x=1:N，其中 N 为向量 Y 的长度，或者为矩阵 Y 的行数。

yi=interp1(x,Y,xi,method)：用指定的算法计算插值。nearest 为最近邻点插值，直接完成计算；linear 为线性插值（默认方式），直接完成计算；spline 为三次样条函数插值。

yi=interp1(x,Y,xi,method,'extrap')：对于超出 x 范围的 xi 中的分量将执行特殊的外插值法 extrap。

yi=interp1(x,Y,xi,method,extrapval)：确定超出 x 范围的 xi 中的分量的外插值 extrapval，其值通常取 NaN 或 0。

3 遗传算法

3.1 概述

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。

3.2 在本问题中的应用

1.编码：在本问题中，每个样品有 51 个点，每个点只有两种状态—取或不取。所以将每个染色体编码成 51 位的二进制码，每一位代表一个点，0 表示不取，1 表示取。

2.解码：将染色体翻译成对应的取点方式。

3.种群：在本问题中将 N（我们选取的为 50）个候选解组合成一个集合，种群是每次迭代时的操作对象，每次迭代称为一代。

4.交叉：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。这里使用的是单点交叉，随机生成一个 0 和 51 之间的数，以该点为中心交换染色体片段。

5.变异：对某些基因进行变动。具体操作是遍历每个基因，并生成一个 0 到 1000 的随机数，如果小于 15（即变异几率为 1.5%）则将该基因由 0 变 1 或由 1 变 0。变异主要是提高算法的全局搜索能力。

6.适应度：这里我们把每种取点方式的定标总成本作为该候选解的适应度。在这里成本越小越优先选取。

7.选择：从种群中根据适应度的大小进行选择，选择出优胜个体，淘汰掉劣质个体。原来有 50 个个体，交叉后再产生 50 个个体，在中间挑选前 50 个作为下一代种群。该操作在实现中并入了交叉中。

3.3 在 matlab 中的实现

代码见附录

3.4 流程表示

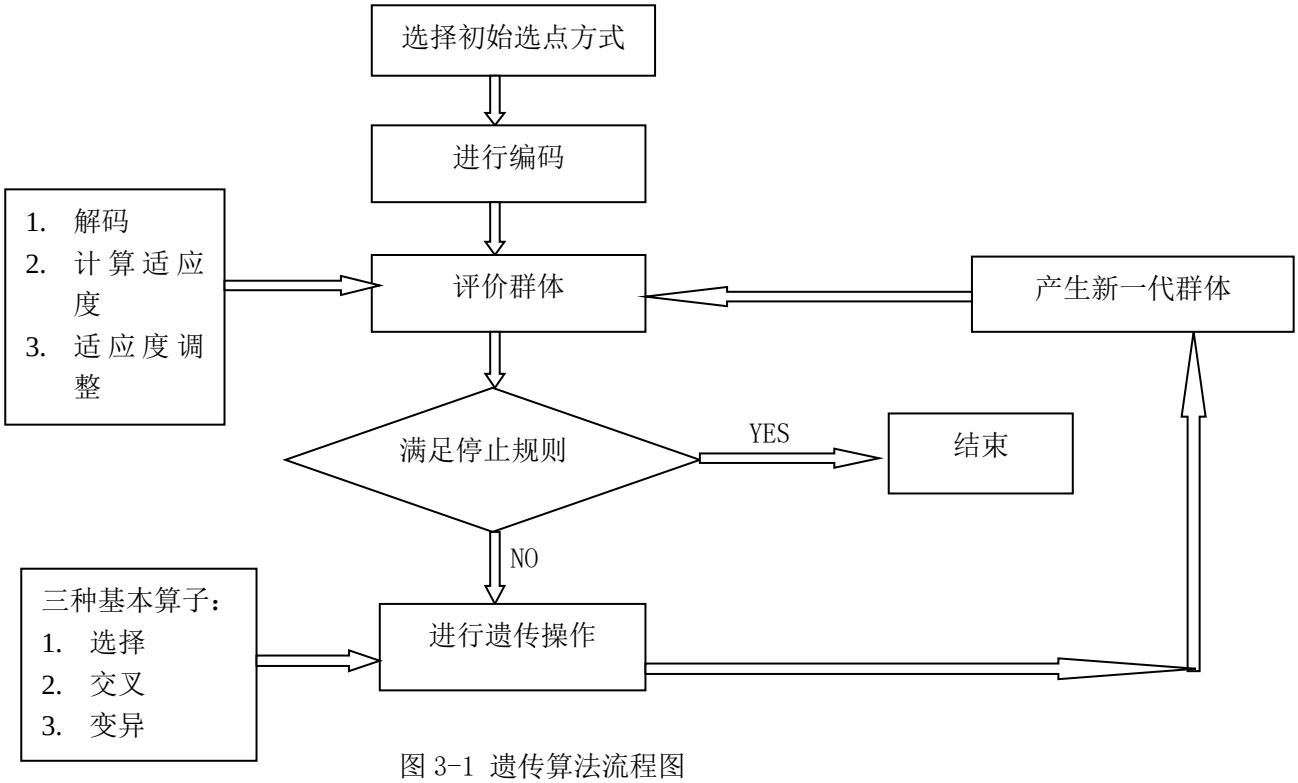

图 3-1 遗传算法流程图

3.5 计算结果

最小成本为：93.55757

取点方式为：3 12 23 32 43 50

拟合方式： 三次样条插值

3.6 误差分析

1.直观分析。由图可知，单点误差基本集中在-1 和 1 之间，误差近似服从均值为 0 的正态分布。在-3 和 3 以外的点数几乎为 0.

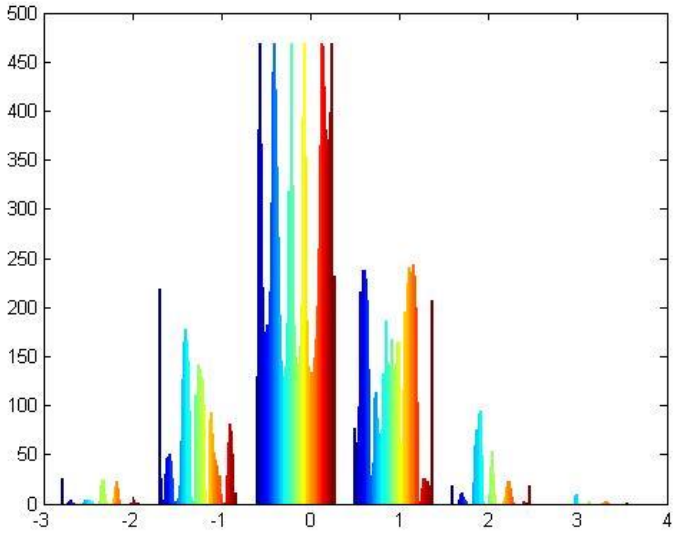

图 3-2 单点误差的分布直方图

2.均值分布。由图可知，不同样品间误差均值存在差异，但都基本分布在-0.4 和 0.4 之间，集中分布在-0.1 和 0.1 之间，计算所得各个样品均值的均值为 0.0258，约等于 0。

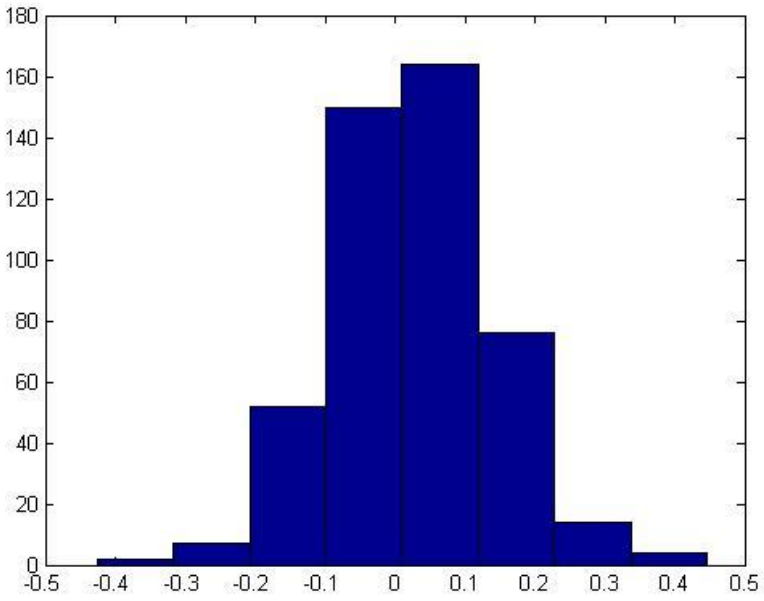

图 3-3 各个样品误差均值的分布直方图

3.标准差分布。由图可知，不同样品间误差标准差存在差异，但都集中分布在 0.6 和 0.8 之间，且最大标准差不超过 1.3。计算得不同样品间误差标准差的均值为 0.6978。

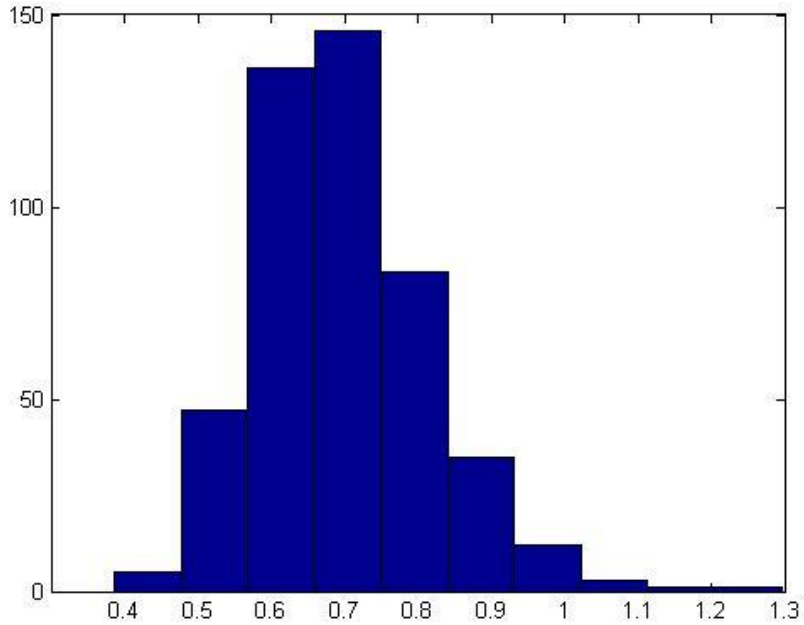

图3-4 各个样品误差标准差的分布直方图

3.6 统计假设以及假设检验

统计假设：在这种取点方式下单点误差分布服从均值为 0，标准差为 0.7 的正态分布。

1. 在标准差未知的情况下检验均值是否为 0，使用 t 检验法（ttest）。

   [h,p,muci,stats]=ttest(x, 0,0.05) 其中 0.05 为显著水平。

   结果为：如图，h 为 0（可以接受原假设）的样本数约为 430 组为 1（不能接受原假设）的样本数约为 30 组。所以可以认为均值为 0。

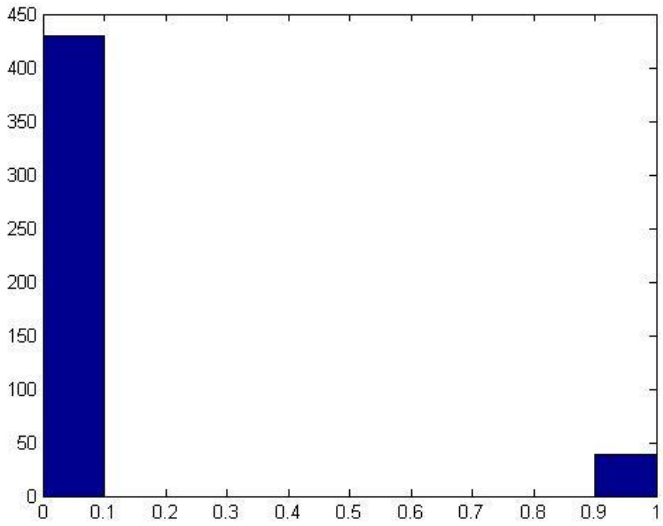

图3-5 均值检验h值的分布

2. 在均值未知的情况下检验标准差是否等于 0.7。

   [h,p,varci,stats]=vartest(x,0.7,0.05) 其中 0.05 为显著水平。

   结果为：如图，有将近 360 个样本 h 值为 0，表示大部分情况下是满足的。

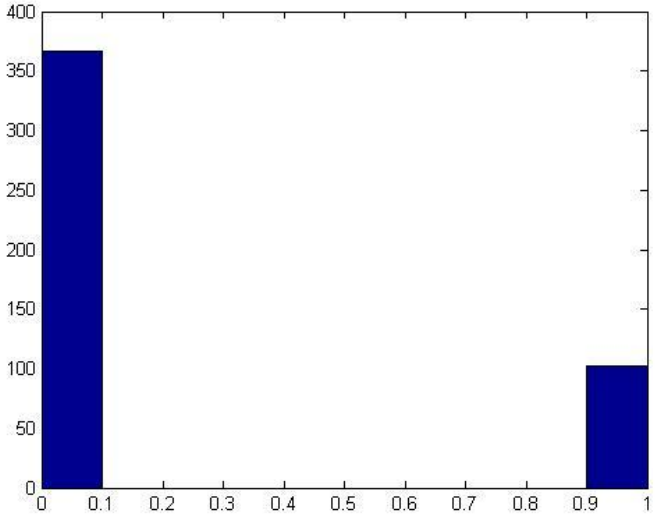

图3-6 方差检验时h值的分布

结论：对于多数样本而言，误差近似满足均值为 0 标准差为 0.7 的正态分布。

4 结论

在这个课题中，对于样本输入输出关系研究和拟合中，我们通过遗传算法和三次样条插值法成功找出一种拟合取点方式—取 6 个点分别位于 3 12 23 32 43 50 的位置。按上述取点方式，平均定标成本大约在 93 左右，在可以接受的范围以内。通过对于误差的统计分析，我们发现误差的分布近似的满足均值为 0 且标准差为 0.7 的正态分布，并且通过参数假设检验成功的验证了其合理性。

5 参考文献

[1] 上海交大电子工程系 . 统计推断在数模转换系统中的应用课程讲义 [EB/OL].ftp://202.120.39.248.

[2] 曹弋 《matlab 教程及实例》 机械工业出版社 2008

[3] 雷英杰 等编著 《matlab 遗传算法工具箱及应用》西安电子科技大学出版社 2005-4-1

6 附录

代码：

控制台输入：

```
>> global minput;
>> minput = xlsread('data');
>>go;
```

1. go 函数:

```
population = ge_init();
fprintf('\n第%i代最小成本为: \n',0)
[population,select] = ge_cross(population);
for i=1:1:100;
fprintf('\n第%i代最小成本为:\n',i);
[population,select] = ge_cross(population);
population=ge_mutation(population);
end
```

2. ge_init 函数：

```
function[population] = ge_init()
population = zeros(50,51);
for i=1:1:51
for j=1:1:51
    tem = unidrnd(51);
    if tem<=5
        population(i,j)=1;
    else
         population(i,j)=0;
  end
end
```

```
    end
```

3. ge_cross 函数：

```
function[result,select] = ge_cross(population)

result = zeros(50,51);
c = zeros(100);
for i=1:1:50
    tem = Cost(population(i,:));
    c(i) = tem;
end
sons = zeros(50,51);
for i=1:1:25

[sons(i,:),sons(50-i+1,:)]=cross(population(i,:),population(50-i+1,:)
);
end
for i=1:1:50
    tem = Cost(sons(i,:));
    c(i+50) = tem;
end
select = zeros(50);
position = zeros(50);
for i=1:1:50
    select(i)= inf;
    position(i)=i;
end
for i=1:1:100
    for j=1:1:50
        if c(i)<select(j)
            for k=50:-1:(j+1)
                select(k)=select(k-1);
                position(k)=position(k-1);
            end
            select(j) = c(i);
            position(j)=i;
            break
        end
    end
end
for i=1:1:50
    if position(i)<=50
        result(i,:) = population(position(i),:);
    else
        y= position(i)-50;
```

```
        result(i,:) = sons(y,:);
    end
end
fprintf('%i\n',select(1));
if(position(1)<=50)
    y =population(position(1),:);
    for i=1:1:51
        if y(i)==1
            fprintf('%i ',i);
        end
    end
else
    y =sons(position(1)-50,:);
    for i=1:1:51
        if y(i)==1
            fprintf('%i ',i);
        end
    end
end4
```

4. ge_mutation 函数：

```
function[result]= ge_mutation(population)

for i=1:1:50
    for j=1:1:51
    tem = unidrnd(1000);
    if(tem<10)
        if population(i,j)==0
            population(i,j) = 1;
        else
            population(i,j) = 0;
        end
    end
    end
end
result = population;
```

5. cross 函数：

```
function[result1,result2] = cross(sample1,sample2)
position = unidrnd(50);
result1 = sample1;
result2 = sample2;
for i=position:1:51
    result1(i)=sample2(i);
    result2(i)=sample1(i);
end
```

6. Cost 函数：

```
function[cost] = Cost(sample)
count =0;
for i=1:1:51
    if sample(i)==1
        count = count+1;
    end
end
if count<=4
    cost=inf;
else
my_answer= zeros(1,count);
count =0;
for i=1:1:51
    if sample(i)==1
        count = count+1;
        my_answer(count)=i;
    end
end
my_answer_n=size(my_answer,2);

global minput;
[M,N]=size(minput);
nsample=M/2; npoint=51;
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;


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


Q=12;
```

```
errabs=abs(y0-y1);
le0_5=(errabs<=0.5);
le1_0=(errabs<=1);
le2_0=(errabs<=2);
le3_0=(errabs<=3);
le5_0=(errabs<=5);
g5_0=(errabs>5);
sij=0.5*(le1_0-le0_5)+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;


end
```

6. mycurvefitting函数：

```
function y1 = mycurvefitting( x_premea,y0_premea )

x=[5.0:0.1:10.0];

y1=interp1(x_premea,y0_premea,x,'spline');

end
```