统计推断在数模转换系统中的应用

组号：06 姓名：鞠灏 学号：5140309127，姓名：隋金晨 学号：5140309128

摘要：运用统计推断的方法解决了传感器定标的优化问题。通过对多项式拟合法、三次样条插值法产生的结果的分析，选择效果比较好的三次样条插值法，并采用遗传算法选取特征值进行实际计算。通过编程计算确定出了较优化的定标方案。
关键词：定标、多项式拟合、三次样条插值、遗传算法。

Application of Statistic Inference in AD&DA Inverting System

ABSTRACT: Statistical inference was used to optimize the sensor calibration. Characteristic values were selected by genetic algorithm (GA).Optimized calibration scheme was obtained by comparing the results calculated by the polynomial fit algorithm and cubic spline interpolation.

Key words: interpolation；polynomial fitting；the cubic spline interpolation；Genetic Algorithm

1. 引言

假定有某种型号投入批量试生产的电子产品，其内部有一个模块，它的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。本课题的要求是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。

1.1 问题解读

结合我们课上所讲的弹簧测力计的案例，我们可以与本次实验所要解决的案例相类比。在制作弹簧测力计时，我们通过测量几个标准被称物，对弹簧测力计进行定标。定标后得到的弹簧测力计部分示数并不是真正的拉力，只是对弹簧测力计受到的拉力的预估值。同理，在本实验中也要通过取部分点 x-y 的值拟合出估测的物理量 y 随示数 x 的变化曲线，通过曲线上的点读出 y 的估测值。而 y 的估测值与 y 的实际值之间存在误差。我们要做的工作就是找出一种较好的方法，在保证 y 的误差尽量小的情况下，使测量的点也尽量得少。

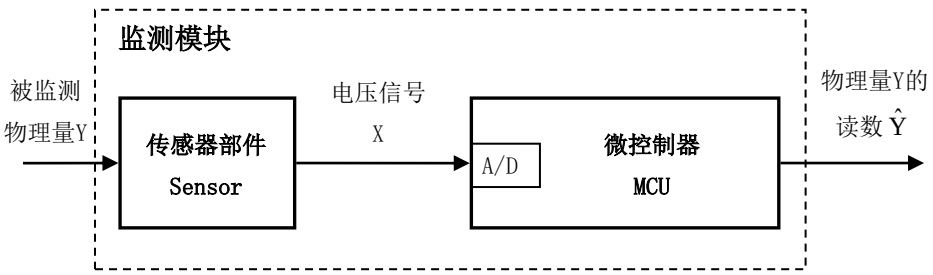

图 1-1 监测模块组成框图[1]

监测模块的组成框图[1]如图 1-1 所示。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）。

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[1]。

1.2 评价标准

为了确定设计方案的好坏，故需选取评价标准。本评价体系由两部分组成：样本单点定标误差成本和样本单点测定成本。

2. 具体分析

本实验中一共提供了 400 个样本，每个样本中有 51 组数据，x 为从 5.0 到 10.0，并且间隔为 0.1 的值，y 为 0 到 100 之间随 x 的增加单调递增的值。传感特性如图 2-1 所示。

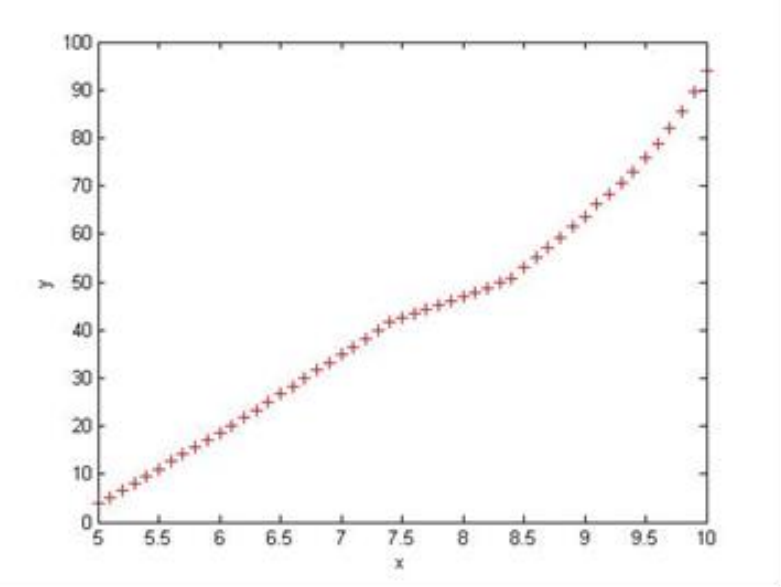

图 2-1 传感特性图示[1]

在图 2-1 中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[1]。

3. 成本计算[1]

为了便于评估及不同的校准方案的比较，特制定以下成本计算规则。

3.1 单点定标误差成本

单点定标误差的成本计算公式如（3-1）所示。

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

单点定标误差的成本按式（3-1）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。

3.2 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 $q=12$。

3.3 某一样本个体的定标成本

样本 i 的定标成本计算如公式 3-2 所示。

$$S_i=\sum_{j=1}^{51}s_{i,j}+q.n_i \quad (3\text{-}2)$$

对样本 i 总的定标成本按式（3-2）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。

3.4 校准方案总成本

按式（3-3）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。

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

总成本较低的校准方案，并认定为是较优的方案。

4. 拟合方案

4.1 三次样条插值

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。样条曲线是由分段三次曲线并接而成，在连接点上二阶导数连续[2]。

定义：函数 $S(x)\in C^2[a,b]$，且在每个小区间 $[x_j,x_{j+1}]$ 上是三次多项式，其中 $a=x_0\prec x_1\prec\cdots\prec x_n=b$ 是给定节点，则称 $S(x)$ 是节点 $x_0,x_1,\cdots,x_n$ 上的三次样条函数。

若在节点 x_j 上给定函数值 $y_j=f(x_j)(j=0,1,\cdots,n)$，并成立

$$S(x_j)=y_j.(j=0,1,\cdots,n) \quad (4\text{-}1)$$

则称$S(x)$为三次样条插值函数。

4.2 多项式拟合

多项式拟合顾名思义是将曲线假定为多项式函数进行拟合计算。因为一次多项式和二次多项式拟合效果并不理想，五次以上多项式拟合则需测量的特征点太多，成本较高。我们初步打算使用三次多项式或四次多项式进行拟合。

4.3 优缺点分析及最终选择

多项式拟合虽然操作简单，易于理解，但是其拟合精度难以有很好的保证，平均误差较大。而三次样条插值虽然计算量大，操作缓慢，但在定标情况下，如果不十分关注定标所用时间，则其拟合曲线光滑，通过所取每一个观测点，与真实曲线符合程度好。三次样条插值计算的实际效果要好于多项式拟合，故最终采用三次样条插值的方法。

5. 特征值选取

在选取特征值时，我们可以选用多种方法，如穷举法、遗传算法等。考虑到本题运算量较大，若选取 6 个点以上至少需要C_{51}^{6}次计算，显然是极为浪费时间的。故我们需要用更加优化的算法来完成特征值的选取。通过查阅资料，我们最终决定使用遗传算法。

6. 遗传算法[3]

遗传算法（Genetic Algorithm，简称 GA）是生命科学与工程科学互相交叉、互相渗透的产物，它是模拟达尔文生的遗传选择和自然淘汰的物进化论一种计算模型，是一种通过模拟自然进化过程搜索最优解的方法。它也可以被看成是对潜在解空间的一种优化搜索过程，它能在搜索过程中自动获取和积累有关搜索空间的知识，并自适应地控制搜索过程以求得最优解。

6.1 遗传算法过程

遗传算法是从代表问题可能潜在解集的一个种群开始的，一个种群由经过基因编码的一定数目的个体组成，初始种群产生之后，按照适者生存和优胜劣汰的原理，逐步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大小挑选个体，并借助自然遗传学的遗传算子进行交叉和变异，产生出代表新的解集的种群。这个过程将导致种群向自然进化一样的后代种群比前代更加适应环境，末代种群中的最优个体经过解码，可以作为问题近似最优解。具体过程如下：

(1)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。

(2)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。

(3)繁殖

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”。

(4)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。

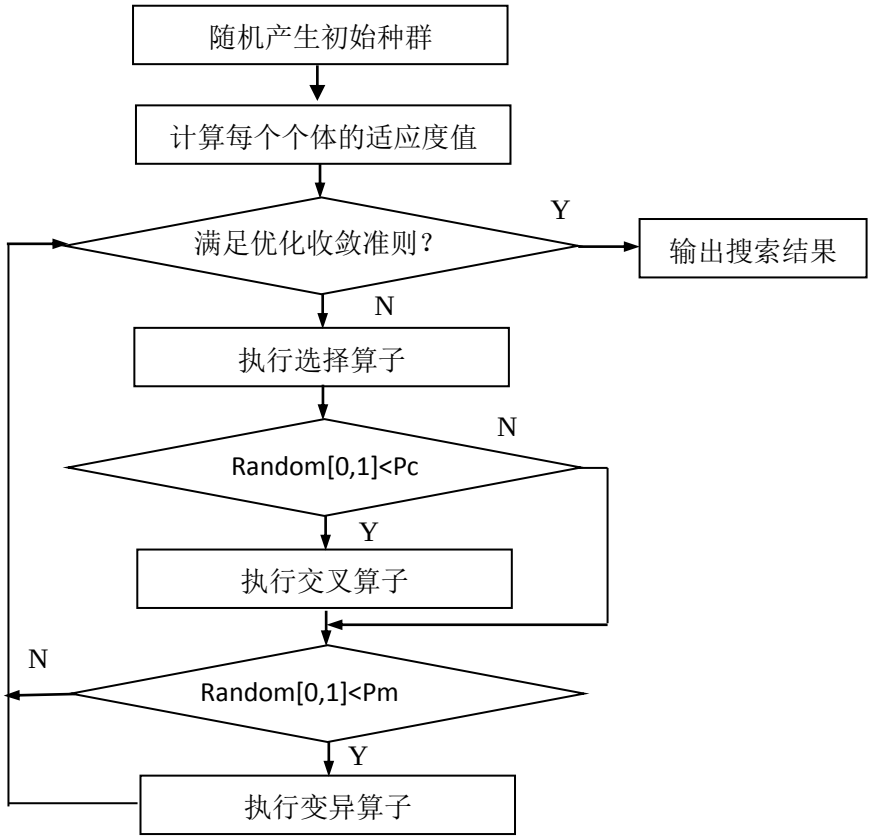

图6-1 遗传算法流程图[3]

6.2 遗传算法的基本运算过程

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（取或者不取）。所以将每个染色体编码成51位的二进制编码，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50个待处理的解组合成一个集和，种群是每次迭代处理时的操作对象，每次迭代之后得到新一代，本问题中，种群为50X51，元素为0，1的矩阵。

4.交叉（繁衍）：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。本问题中采用单点交叉法，随机生成一个0到51之间的数，以该点为中心交换染色体片段。

5.变异：对个别基因进行变动，具体操作是遍历每个基因，并且同时生成一个0到1000的随机数，如果小于10（即变异几率为0.01）则发生变异，即将该基因从0变为1或者从1变为0，用以提高算法的全局搜索能力。

6.适应度：本问题中把每种定标总成本作为该候选解的适应度，定标总成本的计算方法由问题本身已给出，成本越小优先级越高。

7.选择：模拟自然选择优胜劣汰的过程，从中群众根据适应度的大小进行选择，选出优胜个体，淘汰掉劣质个体。原来有50个个体，繁衍产生新的50个个体，从中按优先级选出50个作为新的种群。

7. 结果分析

在本问题中，设置种群大小为 50，利用遗传算法和三次样条插值法，迭代次数为 100，设置不同的变异概率，多次运行程序，得到以下结果。

表 7-1 测定结果

变异概率	取点方案	成本总计
0.1	3 8 22 30 41 50	95.6
0.1	4 8 20 26 33 44 50	95.4
0.1	2 10 20 27 35 44 50	94.1
0.15	2 10 21 29 34 43 50	95.3
0.15	2 10 21 31 43 50	93.7
0.15	3 8 21 29 35 44 50	95.9

由上表可知取 6 或 7 个特征点均可得到较好方案，最后比较成本得到定标方案[2,10,21,31,43,50]作为最后接受的定标方案，最小成本为 93.7。

参考文献

[1]“统计推断”课程设计的要求 V2.2 2015-9-22 ftp://202.120.39.248
[2]百度百科 词条“三次样条插值”
[3]百度百科 词条“遗传算法”

附录

1、 拟合运算

（1）拟合曲线代码实现

```
x=[5.0:0.1:10.0];
y0=xlsread( ‘D:\20150915dataform.xlsx’,’ Sheet1’,’ A2:AY2’,’ basic’ );
p1=polyfit(x,y0,1);
y1=polyval(p1,x);
p2=polyfit(x,y0,2);
y2=polyval(p2,x);
p3=polyfit(x,y0,3);
y3=polyval(p3,x);
p4=polyfit(x,y0,4);
y4=polyval(p4,x);
p10=polyfit(x,y0,10);
y10=polyval(p10,x);
xx=[5:0.05:10];
y5=interp1(x,y0,xx,’ cubic  spline’ );
```

（2）拟合图像

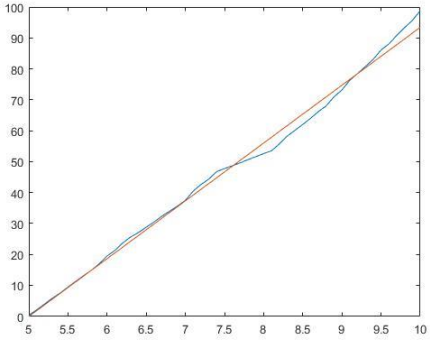

图 1.一次多项式拟合与实际曲线对比

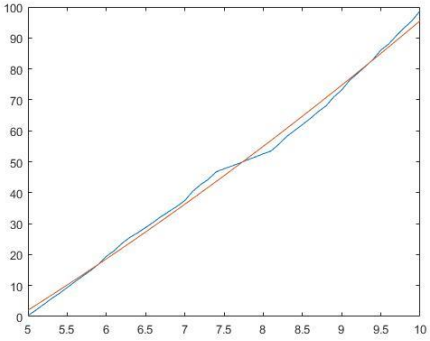

图 2. 二次多项式拟合与实际曲线对比

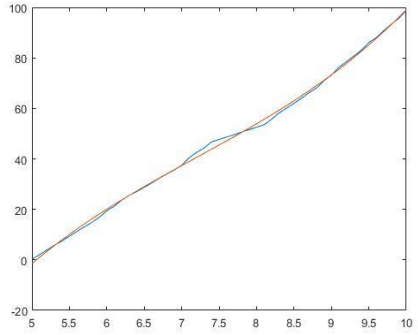

图 3.三次多项式拟合与实际曲线对比

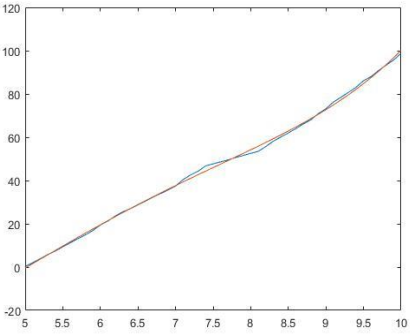

图 4. 四次多项式拟合与实际曲线对比

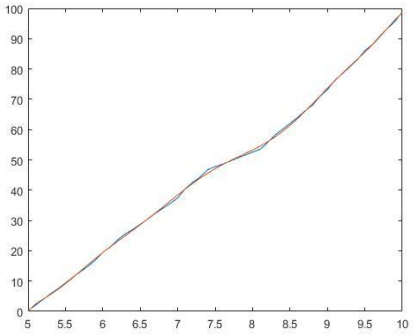

图 5.十次多项式拟合与实际曲线对比

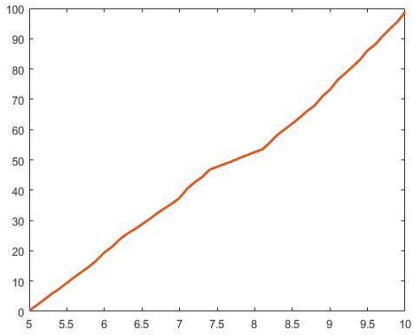

图 6.三次样条插值与实际曲线对比

通过 Matlab 拟合我们得到一次多项式、二次多项式、三次多项式、四次多项式、十次多项式以及三次样条插值与实际曲线的对比图像。可以看出，一次和二次多项式拟合很不理想，三次和四次多项式拟合误差在可接受范围内，十次多项式拟合以及三次样条分析拟合情况比较完美。但考虑到十次多项式拟合测定成本过高，难以得出较优化结果。

最终，我们在继续测试后，在三次多项式拟合、四次多项式拟合以及三次样条插值中选择了三次样条插值的拟合方案。

2.遗传算法

Go 函数

```
global minput
minput = xlsread( ‘20150915dataform.csv’ )
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
```

设定初始种群

```
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
```

交叉函数（繁衍），并且选择下一代种群。

```
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
end
```

种群变异

```
function[result]= ge_mutation(population)
for i=1:1:50
for j=1:1:51
tem = unidrnd(1000);
if(tem<15)
if population(i,j)==0
population(i,j) = 1;
else
population(i,j) = 0;
end
end
end
end
end
result = population;
```

选择函数子函数，成本计算

```
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
```

拟合方案，三次样条差值拟合

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

```
x=[5.0:0.1:10.0];
y1=interp1(x_premea,y0_premea,x,'spline');
end
```