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 15 组，成员 王延之 学号 5140309352, 李亚旻 学号 5140309209, 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	《统计推断在数模模数转换中的应用》，徐鹏飞，2014 年秋季学期，组号 16 在该组报告附录提供的程序代码基础上，进行了少量修改。
拟合方法确定方面	《统计推断在数模模数转换中的应用》，沈思齐，2014 年秋季学期，组号 01 “对样本数据进行观察”部分参考了该组报告的观察方法和描述。
算法思路	《统计推断在数模模数转换中的应用》，徐鹏飞，2014 年秋季学期，组号 16 参考了该组思路

除了以上注明的参考内容，和报告中列出的引用文献注解，本报告其他部分都不包含任何其他个人或集体已经发表或撰写过的作品成果。

统计推断在模数、数模转换系统中的应用

第 15 组 王延之 5140309352， 李亚旻 5140309209

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。文中通过分析不同拟合方法，并使用遗传算法，为某电子产品其中一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。拟合与遗传算法通过 Matlab 实现。

关键词：多项式拟合，三次样条插值，遗传算法，Matlab

Application of Statistical Inference in AD&DA Inverting System

ABSTRACT: This is the report of 'Application of Statistical Inference in AD&DA Inverting System' designed by SEIEE,SJTU.By analyzing different fitting methods as well as using the principle of Genetic Algorithm，we can design a scheme of sensing characteristics calibration with reasonable cost for the volume production of a electronic product's module.The software Matlab can realize the fitting methods and the principle of Genetic Algorithm.

Key words: Polynomial fitting,Cubic spline,Genetic Algorithm,Matlab

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 基本问题

在工业生产中，我们往往需要通过传感器测量相关的参数，但是传感器使用种种原理将带测量转换为直接可观测的量的过程中往往要经过一系列的非电信号到电信号转换与机械传动，因此被检测物理量与直接测出量之间绝大数时候并不呈线性，当非线性带来的误差不能被接受的时候就需要重新定标。然而对于器件一致性差，样本容量大的情况，传统的密集选点法并不能高效地完成校准定标的工作，并且在测量中将付出极大的成本，这就要求我们寻求更为优化的方法完成校准定标的工作。[1]

2 评价标准

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

$$s_{i,j}=\begin{cases}0 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|\le 0.4\\ 0.1 & \text{if } 0.4<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 0.6\\ 0.7 & \text{if } 0.6<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 0.8\\ 0.9 & \text{if } 0.8<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 1\\ 1.5 & \text{if } 1<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 2\\ 6 & \text{if } 2<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 3\\ 12 & \text{if } 3<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 5\\ 25 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|>5\end{cases} \tag{2.1}$$

单点定标误差的成本按式（2.1）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

$$S_i=\sum_{j=1}^{51}s_{i,j}+q\cdot n_i \tag{2.2}$$

对样本 i 总的定标成本按式（2.2）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（2.2）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。

$$C=\frac{1}{M}\sum_{i=1}^{M}S_i \tag{2.3}$$

总成本较低的校准方案，认定为较优方案。

3 拟合方法的确定

3.1 对样本数据进行观察

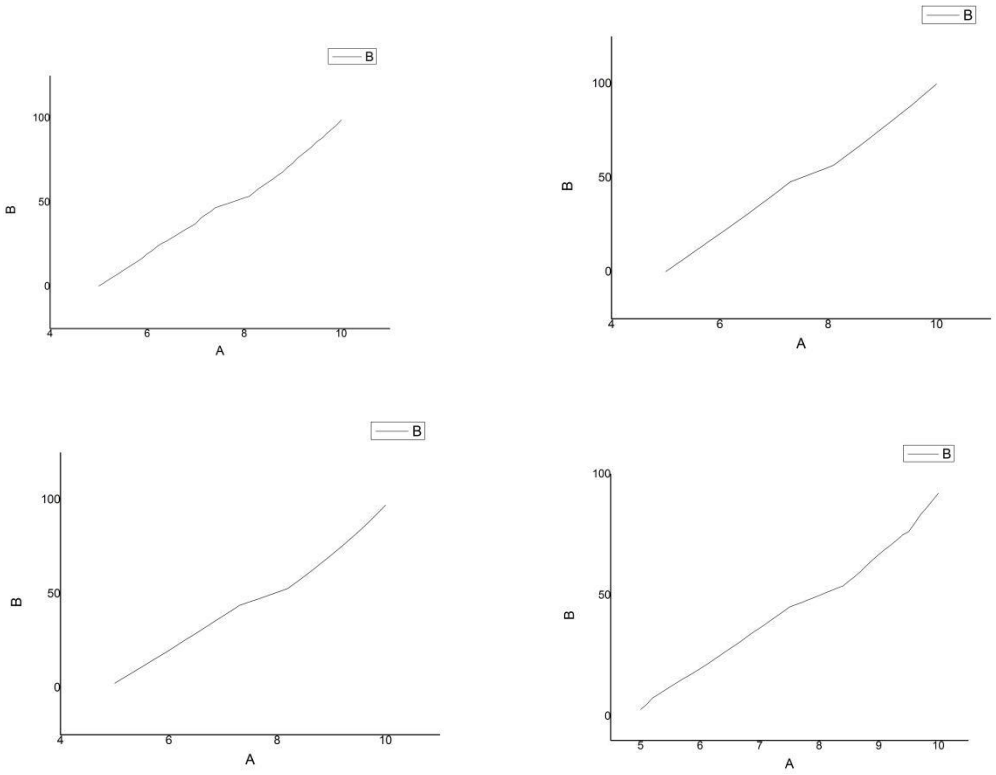

图 3.1.1 样本 1 120 250 330 特性图示对比

可以看出其特征：

- 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。

3.2 不分段拟合

将每组数据视作整体进行拟合。因为希望运用尽量简单的函数进行拟合，在对样本数据的观察中我们得出“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”的结论，故无法采用线性函数或指数函数等简单函数进行拟合。而多项式函数可以做到“简单”和“斜率有三部分”兼而有之。

在多项式函数中，由于斜率分为三段，首先可以排除用二次函数拟合。而另一方面，取点方案如果取出较多的点将会增大成本，取点个数在 7 个左右是我们的理想状态。故而七次及以上次数的函数在拟合中由于样本点较少而不能准确表达。

基于以上的考虑，我们最终选用形式简单、拟合较准确的三次函数和五次函数进行不分段拟合。

3.3 分三段进行拟合

每个图像均有明显的三部分，故可以分三段并使用三次样条插值法。

三次样条插值是三次样条插值法是数学上的一种拟合方法。从数学角度分析，在每个区间

$[X_k, X_k+1]$可 构造一个三次函数，使得分段曲线 y = S(x)和它的一阶导数和二阶导数在更大的区间$[X_0, X_n]$ 内连续。S′ (x)的连续性意味着曲线 y = S(x)没有急弯。S′ ′ (x)的连续性意味着每点的曲率半径有定义。

而 Matlab 中自带三次样条插值函数 spline 用于拟合。

4 方案选择

4.1 算法选择

4.1.1 暴力穷举

经过简单计算，完全通过穷举选取极为复杂，与我们的初衷也相违背。故暴力穷举不可取。

4.1.2 遗传算法[2]

遗传算法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，可以作为问题近似最优解。

优点：质量高，初值鲁棒性强，简单、通用、易实现。

缺点：(1)单一的遗传算法编码不能全面地将优化问题的约束表示出来。考虑约束的一 个方法就是对不可行解采用阈值，这样，计算的时间必然增加。

(2)遗传算法通常的效率比其他传统的优化方法低。

(3)遗传算法容易过早收敛。

(4)遗传算法对算法的精度、可行度、计算复杂性等方面，还没有有效的定量分析方法。

遗传算法的基本运算过程如下：

（1）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)。在我们的 inherit 函数中，T 与 M 分别由变量 gen 和 num 表示，且 gen 取 100，num 取 100-200。

（2）个体评价：计算群体 P(t)中各个个体的适应度。

（3）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。我们使用了赌轮法进行选择。

（4）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。

（5）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。在 inherent 函数中用 rate 表示变异率，并取 rate=0.01。

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)。

（6）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。

4.2 算法思路

把每个取点方案用 51 位二进制数对应的一维数组表示。通过适应度函数计算种群中每种基因型的适应度及生存概率,然后用赌轮法筛选个体。之后进行交叉互换，把 51 个基因分为 2 段,两两随机配对得到新个体。之后随机取若干个位置,令这这些位置的基因发生变异，得到最终的第二代个体可能的基因型。之后再重复相同的操作直至循环代数足够多，循环结

束过程中生存概率最大的基因型对应的 51 位二进制数对应的取点方案是该种拟合较好的取点方案。

4.3 算法实现上出现的问题及应对

4.3.1 出现的问题

主要问题有两个，分别是可能出现种群退化，还有在进化后期由于在 51 个点上有四十余个零，大量的交换是无效的，结果陷入局部最优解，无法得到最优结果。

4.3.2 解决方案

（1）适应度函数的选取

适应函数一般可选区成本函数的倒数，但在本问题实际操作时出现了种群进化慢的问题。并且在进化后期容易陷入局部最优解。我们的解决方案是先将成本函数减去适当的值再求倒数。而我们选取的三种拟合方法（三次函数、五次函数、三次样条插值）所需的样本点）都不会少于四个，故可以先将成本函数减去 4*12=48，再求倒数，以之为适应度函数，即

$$live(u)=1000/(t-48) \quad (4.1)$$

（2）对种群退化的应对

为防止种群退化的出现，我们人为保留了每一代最优秀的个体。

（3）陷入局部最优解的应对

当最优解停留在某一值大于 10 代时，为防止该值仅为局部最优解，对此解采取交换片段结合的方法以得到更优的个体。本算法交换了最优解随机产生的位点左右两侧的基因，考虑到运行至后期，约有 6/7 的点左右两边都是 0，故每次产生 7 个位点，使操作一次约进行 1 次有效片段交换。

5 结论

5.1 数据结果

对于三种选取的拟合方法，分别得到其最优解。

三次样条插值法所得成本最小的取样方法为六个点[1，11，22，31，42，50]，所得成本为 97.04。

三次函数拟合法所得成本最小的取样方法为五个点[5,15,27,38,48]，所得成本为 116.20。

五次函数拟合法所得成本最小的取样方法为七个点[4，11，21，29，39，47，51]，所得成本为 112.90.

5.2 数据分析

对三种拟合方法最小成本进行分析，三次样条插值法成本最小，而三次函数拟合所得成本最大。

对三种拟合方法计算用时进行分析，三次样条插值法用时亦是最小的，五次函数拟合用时最长。

5.3 最终结论

通过对三种拟合方法的比较，最终可以得出，使用三次样条插值法为最优方法。

在本问题中，取点方案为[1，11，22，31，42，50]，成本为 97.04。

6 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] “遗传算法”词条，百度百科
[3] matlab_tutorial，由上海交大电子工程系提供
[4] mycurvefitting、test_ur_answer 函数，有上海交大电子工程系提供

1 f3 函数

```
function z=f3(n) %n 为定标所测的点的序号构成的一维数组,输出为以样条插值函数为拟合函数的最后的平均成本%
M=csvread('20150915dataform.csv');
S=0;
x=zeros(400,51);
y=zeros(400,51);
f=zeros(400,51);   %拟合计算得到的y 值
N=length(n);
for j=1:400
x(j,:)=M(2*j-1,:);
y(j,:)=M(2*j,:);
end
X=x(:,n);
Y=y(:,n);
for j=1:400
f(j,:)=interp1(X(j,:),Y(j,:),x(j,:),'spline');
end
err=abs(f-y);
a1=(err<=1 & err>0.5);
a2=(err<=2 & err>1);
a3=(err<=3 & err>2);
a4=(err<=5 & err>3);
a5=(err>5);
S=a1*0.5+a2*1.5+a3*6+a4*12+a5*25;
C=sum(sum(S,2))/400+12*N;
```

2 fitting函数

```
function z=fitting(n) %n 为定标所测的点的序号构成的一维数组,输出为以三次多项式函数为拟合函数的最后的平均成本%
M=csvread('20150915dataform.csv');
S=0;
x=zeros(400,51);
y=zeros(400,51);
p=zeros(400,4);
f=zeros(400,51); %拟合计算得到的y 值
N=length(n);
for j=1:400
x(j,:)=M(2*j-1,:);
y(j,:)=M(2*j,:);
end
X=x(:,n);
```

```
Y=y(:,n);
for j=1:400
p(j,:)=polyfit(X(j,:),Y(j,:),3);
for k=1:51
f(j,k)=p(j,1)*(x(j,k))^3+p(j,2)*(x(j,k))^2+p(j,3)*(x(j,k))+p(j,4);
end
end
err=abs(f-y);
a1=(err<=1 & err>0.5);
a2=(err<=2 & err>1);
a3=(err<=3 & err>2);
a4=(err<=5 & err>3);
a5=(err>>5);
S=a1*0.5+a2*1.5+a3*6+a4*12+a5*25;
C=sum(sum(S,2))/400+12*N;
```

3 f5 函数

```
function z=f5(n) %n 为定标所测的点的序号构成的一维数组，输出为以五次多项式函数为拟合函数的最后的平均成本%
M=csvread('20150915dataform.csv');
S=0;
x=zeros(400,51);
y=zeros(400,51);
p=zeros(400,6);
f=zeros(400,51); %拟合计算得到的y 值
N=length(n);
for j=1:400
x(j,:)=M(2*j-1,:);
y(j,:)=M(2*j,:);
end
X=x(:,n);
Y=y(:,n);
for j=1:400
p(j,:)=polyfit(X(j,:),Y(j,:),5);
for k=1:51
f(j,k)=p(j,1)*(x(j,k))^5+p(j,2)*(x(j,k))^4+p(j,3)*(x(j,k))^3+p(j,4)*(
x(j,k))^2+p(j,5)*(x(j,k))+p(j,6);
end
end
err=abs(f-y);
a1=(err<=1 & err>0.5);
a2=(err<=2 & err>1);
a3=(err<=3 & err>2);
a4=(err<=5 & err>3);
```

```
a5=(err>5);
S=a1*0.5+a2*1.5+a3*6+a4*12+a5*25;
C=sum(sum(S,2))/400+12*N;
```

4 trans 函数

```
function e=trans(o) %将取点的数组转化为51 个1，0 组成的数组
e=zeros(1,51);
e(o)=1;
```

5 transfer 函数

```
function q=transfer(t) %把51 个1 或0 构成的数组转化为取点序号构成的数组
j=0;
q=zeros(1,1);
for i=1:51
if t(i)==1
j=j+1;
q(j)=i;
end
end
```

6 inherit 函数（遗传算法主函数）

```
function inherit(num,gen,rate) %遗传算法,输出为最佳取点方案
%num 为种群数量，gen 为运行代数，rate 为变异率
%定义种群中的num 个初始个体，二维数组a 的每一行代表一个个体
%下面的为三次样条插值拟合，将下面的f3 替换为 fitting 后即为 3 次多项式拟合
tic;
a=zeros(num,51);
b=zeros(num,51);
live=zeros(1,num);
plive=zeros(1,num);
for i=1:num
for j=1:17
h=randi(3)+3*j-3;
a(i,h)=1;
end
end
time=0;
best=a(1,:);
last=best;
bst=transfer(best);
same=0;
%多代繁衍可以得到较优的基因型
while time<gen
time=time+1;
total=0;
```

```
average=0;
%适应度函数，计算适应度和生存概率，得到最优的个体
for u=1:num
cur=transfer(a(u,:));
if(length(cur)<=1)
live(u)=0;
average=average+51*25;
else
t=f3(cur);
live(u)=1000/(t-48);
average=average+t;
end
total=total+live(u);
end
average=average/num;
live=live/total;
[x,s]=max(live);
best=a(s,:);
if f3(transfer(best))<f3((transfer(last)))
last=best;
end
%记录相同最优个体的持续代数
if(best==trans(bst))
same=same+1;
else
same=0;
end
%利用赌轮法进行随机选择
plive=cumsum(live);
for u=1:num
ran=rand;
tmp=find(ran<plive);
b(u,:)=a(tmp(1),:);
end
a=b;
%随机交换a 中个体的顺序，保证下面的配对交叉是随机的
for u=1:floor(num/2)
ran1=randi(num);
ran2=randi(num);
a([ran1,ran2],:)=a([ran2,ran1],:);
end
%配对交叉产生下一代个体
for u=1:floor(num/4)
b(4*u-3,22:51)=a(4*u-2,22:51);
```

```
b(4*u-2,22:51)=a(4*u-3,22:51);
b(4*u-1,31:51)=a(4*u,31:51);
b(4*u,31:51)=a(4*u-1,31:51);
end
a=b;
%随机变异
for uu=1:floor(rate*51*num)
ranx=randi(num);
rany=randi(51);
a(ranx,rany)=1-a(ranx,rany);
end
a(1,:)=best;%保留最优个体，若陷入局部最优解则定向变异
if same>=10
for u=1:7
ran=randi(50);
a(1,[ran,ran+1])=a(1,[ran+1,ran]);
end
ran=randi(51);
a(1,ran)=1-a(1,ran);
end
%输出
diary('output.txt');
if mod(time,10)==0
bst=transfer(best);
fprintf('generation: %d\n',time);
fprintf('best of the generation: %d\n');
disp(bst);
fprintf('value of the best:%d\n');
disp(f3(bst));
fprintf('average of the generation:%d\n');
disp(average);
end
end
fprintf('best of the all the generations: %d\n');
disp(transfer(last));
fprintf('value of the best:%d\n');
disp(f3(transfer(last)));
diary off;
toc;
```

7.mycurvefitting1函数

```
%用于三次样条插值法
function y1 = mycurvefitting1( x_premea,y0_premea )
x=[5.0:0.1:10.0];
```

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

8. test_ur_answer1 函数

```
%用于三次样条插值法
my_answer=[ 1,11,22,31,42,50 ];
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
     y1(j,:)=mycurvefitting1(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);
```

9. mycurvefitting2 函数

```
%用于三次函数拟合
function q = mycurvefitting2( x_premea,y0_premea )

q=polyfit( x_premea,y0_premea,3);

end
```

10. test_ur_answer2 函数

```
%用于三次函数拟合计算成本
my_answer=[ 5,15,27,38,48 ];%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);
q=zeros(nsample,4);
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

    q(j,:)=mycurvefitting2(x_optimal(j,:),y0_optimal(j,:));
for k=1:51
y1(j,k)=q(j,1)*(x(j,k))^3+q(j,2)*(x(j,k))^2+q(j,3)*(x(j,k))+q(j,4);
end
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);
```

11. Mycurvefitting3 函数

```
%用于五次函数拟合
function p = mycurvefitting3( x_premea,y0_premea )

p=polyfit(x_premea,y0_premea,5);

end
```

12. test_ur_answer3 函数

```
%用于五次函数拟合成本计算
my_answer=[ 4,11,21,29,39,47,51 ];%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);
p=zeros(nsample,6);
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

    p(j,:)=mycurvefitting3(x_optimal(j,:),y0_optimal(j,:));
for k=1:51
y1(j,k)=p(j,1)*(x(j,k))^5+p(j,2)*(x(j,k))^4+p(j,3)*(x(j,k))^3+p(j,4)*
(...
x(j,k))^2+p(j,5)*(x(j,k))+p(j,6);
end
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);
```