统计推断在数模转换系统中的应用

——寻求校准工序的优化方案

组号：20　　艾玥祺 5130309128 尹艺媛 5130309129

摘要：本文是《统计推断在数模转换系统中的应用》的课程小论文终稿。基于课题的背景及课程的基本要求，本文首先介绍了如何用 Matlab 寻求实现通过几个点拟合出一条曲线的优化方法；然后重点介绍了如何通过遗传算法进行启发式搜索，解决一个组合优化问题的思路和具体操作；最后汇报成果。

关键词：统计推断，Matlab，数模转换，遗传算法

Application of Statistical Inference in DA Inverting System

ABSTRACT: This report is the final report for the course 'Application of Statistical Inference in DA Inverting System'. Based on the background of the course, this article first introduces how to find a better method to get the fitting function by several points with the help of the software Matlab. It also introduces the idea and specific process of finding the seven distinctive points with the principle of Genetic Algorithm, and shows the solution at last.

Key words: Statistical Inference, Matlab, DA Inverting system, 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]。实验共有 469 组数据，每组数据对应有 51 个点。

1.2 课程基本问题

由于实验数据较多，因此分析处理工作量很大。所以需要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。

2. 寻找基于某一组合的拟合函数

2.1 观察样本数据

通过对一个典型样品的曲线（如图 1）特点的分析我们可以发现:

(1) 特性曲线都是单调递增的；

(2) 特性曲线大致可以分为首（左）、中、尾（右）三段，三段都不是完全线性的，有一定弯曲度（随机），中段的斜率小于首段和尾段的斜率，且中段的起点位置和长度都带有随机性；

(3) 个体产品的特性曲线形态相似但两两相异。因此，可以用不同的拟合曲线来表示不同样品 X-Y 关系。

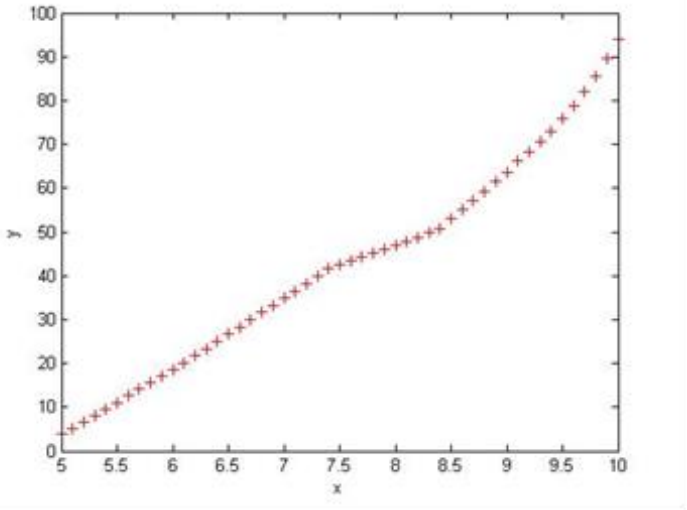

图 1 传感特性图示

2.2 探索不同的拟合方法

2.2.1 多项式拟合法

设一条三次多项式曲线为：

$$y = a_3x^3 + a_2x^2 + a_1x + a_0, \tag{1}$$

随机选择七个点进行多项式拟合，通过某一样本个体的定标成本函数

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

和校准方案总体成本函数

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

计算得出平均成本。计算出的成本如下表所示：

表 1 多项式拟合平均

所选择的 7 个点编号	平均成本
[1 9 17 25 33 41 51]	125.416
[3 10 18 24 34 42 51]	123.115
[2 9 17 24 32 41 50]	120.217

2.2.2 三次样条插值法

我们又尝试了三次样条插值法。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。

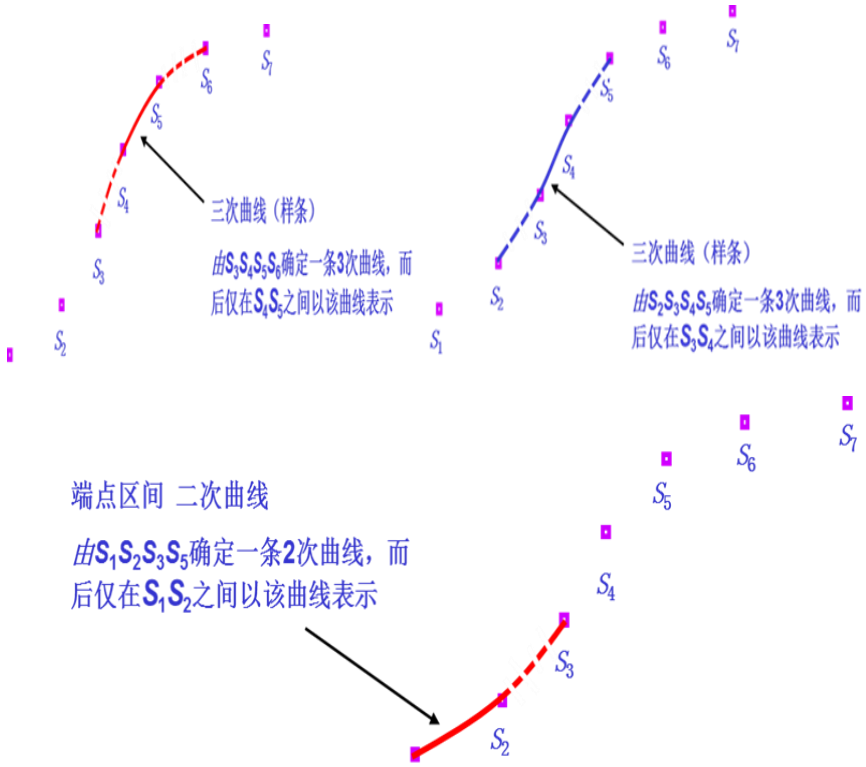

图 2 三次样条插值原理

MATLAB 里有插值函数 spline，可以直接调用实现三次样条插值。通过取与拟合方法相同的三组点，计算成本如下表所示：

表 2 三次样条插值平均成本

所选择的 7 个点	平均成本
[1 9 17 25 33 41 51]	99.49
[3 10 18 24 34 42 51]	99.11
[2 9 17 24 32 41 50]	96.46

2.2.3 结论

通过表 1 和表 2 的对比，发现用三次样条插值法计算得出的成本较低，所以我们决定用三次样条插值法得出表达式。

3. 利用遗传算法解决组合最优化问题

由于要在 51 个点中选取几个点来进行曲线拟合，所以备选的组合方案数量巨大，目前的计算机，尚难以对以上问题采取简单穷举，所以尝试采用“启发式搜索”。我们采用的启发式搜索算法为“遗传算法”。

3.1 遗传算法简介

遗传算法（Genetic Algorithm）[2]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。

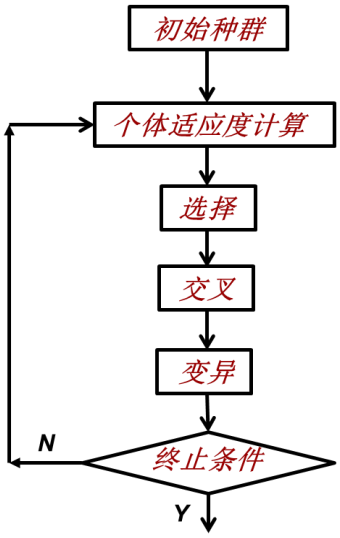

图 3 遗传算法的流程图

遗传算法是解决搜索问题的一种通用算法，对于各种通用问题都可以使用。搜索算法的共同特征为：

(1) 首先组成一组候选解

(2) 依据某些适应性条件测算这些候选解的适应度

(3) 根据适应度保留对候选解作出选择

(4) 对候选解进行某些操作，生成新的候选解

3.2 利用遗传算法的思想确定最优的点组合

由于事先没有确定选择多少个点，所以我们分别尝试了 5 个点，6 个点和 7 个点的情况。下面以 6 个点的情况为例。

3.2.1 遗传算法实现的基本步骤

第一步，产生初始群体。我们确定种群的规模大小为 200，一共选择 6 个点。然后将 1~51 这 51 个点的编号转换成 6 位的二进制编码，因为 6 位的二进制编码转换成十进制最大为 2^6=64 可以表示 51 个点(交叉变异时的超范围问题在后几步中会有讨论)。然后在表示出编号对应的 x 的数值。

第二步，计算适应度。根据选定的 6 个点对应的 y 值，用三次样条插值法拟合出 y 的曲线，将拟合值与测定值相比较计算出总的平均成本。用平均成本倒数的三次方来表示适应度(为使收敛速度加快，经过程序的运行调试，最终用平均成本倒数的三次方作为适应度)，平均成本越小的适应度越高。

第三步，计算选择结果。把当前群体中适应度较高的个体按某种规则或模型遗传到下一代群体中。一般要求适应度较高的个体将有更多的机会遗传到下一代。

第四步，交叉。交叉运算是产生新个体的主要过程，随即设定交叉点的位置，打乱 size 下标。在交叉运算过程中，可能产生不在 1-51 之间的个体，先不做处理，在下一步变异运算时淘汰。

第五步，变异。一共有 6 个点，每个点均可能变异。我们设定变异概率为$\frac{1}{300}$，变异点固定为二进制的最后一位或两位（对应于 10 进制的 1~3），固定变异位置的原因是：收敛之后对每一个数据点以较小的概率作较小的改动，更容易产生最优解；若变异时改动过大，反而会偏离最优解。（在面谈时，我们的存在的问题是当出现超过范围的点时，用当前最优解代替，会导致早熟。在最终方案中，选择随机产生一个属于 1~51 范围内的点来替代超范围的点）

这样，通过以上五步，就能实现用遗传算法来选择一组最优的解。

3.2.2 具体计算及结果

我们用遗传算法分别尝试了选取 5 个点，6 个点和 7 个点时的情况，所得到的最优解的点组合如下表所示：

表 3 不同点数所得成本的比较

点数	选点方案	成本
5	[3 14 26 39 49]	103.562900
6	[3 12 22 31 43 50]	92.877399
7	[2 9 20 26 34 43 50]	93.899787

显然，当选点数较少时，虽然测定成本较低，但是误差成本偏高，即使总成本低也不适用于作为合理的传感特性校准（定标工序）方案。当选点数较多时(大于我们已经尝试过的最多选点 7)，测定成本$\geq 8 \times 12 = 96$，显然不是最优的成本。

3.3 结论

通过多次尝试，我们得出的最佳的选点方案是选择 6 个点分别为[3 12 22 31 43 50]，最优成本为 92.877399。

4. 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__百度百科[EB/OL]

附录

1. 遗传算法主程序（main.m）：

```
data=xlsread('y.xlsx');

%用遗传算法确定最优选点方案（三次样条插值法）
%1.产生初始群体
size=200;                          %规模大小
point=6;                           %点数
D=200;                             %总遗传代数
G=1;                               %当前代数
xp=fcreatexp(size,point);          %随机产生 size 组选点方案(1-51)，每组点数为
                                    point。 [size*point]

while(G<=D)
code=fdtob(size,point,xp);         %将编号 xp 转换成对应的二进制编码
[size*point*6]
x=fx(size,point,xp);               %将 1-51 的选点转换为 x 值 5.0-10.0
[size*point]

%2.计算适应度
y=fy(size,point,data,xp);          %选定的 point 个点对应原始数据的 y 值
[size*469*point]
ys=fspl(size,point,x,y);           %拟合出的 y 标准值 [size*469*51]
w=fw(size,ys,data);                %总平均误差成本 [1*size]
A=fA(size,point,w);                %总平均成本 [1*size]
index=findex(size,A);              %最小值的下标
fprint(A,index,point,xp,G);        %打印当前最优解
live=flive(size,point,A,xp);       %存活概率，即适应度分布区间 [1*size]

%3.计算选择结果
num=fnum(size,live);               %不同个体被选中的次数 [1*size]
select=fselect(size,point,num,code); %选择结果（二进制表示）[size*point*6]

%4.交叉
cross=fcross(size,point,select);   %交叉结果（二进制表示）[size*point*6]

%5.变异
muta=fmuta(size,point,cross);      %变异结果（二进制表示）[size*point*6]
xp=fback(size,point,muta,index,xp); %将二进制转回 10 进制，并修改不符合要求的点

G=G+1;
End
```

2.产生选点方案程序(fcreatexp.m 与 fcreate.m)

```
%fcreatexp.m
%随机选定 point 个点 x_position,产生 size 组数据
function f=fcreatexp(size,point)
x=zeros(size,point);
minn=1; maxn=51;
for i=1:size
    z=fcreate(minn,maxn,point);
    for j=1:point
        x(i,j)=z(j);
    end
end
f=x;
end
%fcreate.m
%产生取值为[minn,maxn]的 n 个随机数，并去重，排序
function f=create(minn,maxn,n)
x=zeros(1,n);
    same=1;
while same==1   %如果有相同的，重新产生随机数
   for i=1:n;
       x(i)=randi(maxn-minn+1)+minn-1;
   end
   same=0;
   for i=1:n-1
        for j=i+1:n
            if x(j)==x(i)
                same=1;
            end
        end
    end
end

for i=1:n-1       %排序
    for j=i+1:n
        if x(j)<x(i)
            t=x(j); x(j)=x(i); x(i)=t;
        end
    end
end
f=x;
end
```

3.把 1-51 编号转成二进制码程序(fdtob.m)

```
%(decimal to binary)将 x 编号转换成对应的二进制编码
```

```
function f=fdtob(size,point,xp)
c=zeros(size,point,6);
for i=1:size
    for j=1:point            %转化为对应的二进制码
        t=xp(i,j);
        for k=6:-1:1
            c(i,j,k)=mod(t,2);
            t=floor(t/2);    %向下取整
        end
    end
end
f=c;
end
```

5. 将选点方案转换为对应的 x 值（fx.m）

```
%将选点方案转换为对应的 x 值
function f=fx(size,point,xp)
x=zeros(size,point);
for i=1:size
    for j=1:point
        x(i,j)=5+(xp(i,j)-1)*0.1;
    end
f=x;
end
```

6. 取出对应原始数据的 y 值（fy.m）

```
%取出对应原始数据的 y 值
function f=fy(size,point,data,xp)
y=zeros(size,469,point);
for i=1:size
    for j=1:469
        for k=1:point
            y(i,j,k)=data(j,xp(i,k));
        end
    end
end
f=y;
end
```

7. 计算插值拟合出的 y 标准值

```
%计算插值拟合出的 y 标准值
function f=fspl(size,point,x,y)
ys=zeros(size,469,51);
for i=1:51
    xd(i)=5+(i-1)*0.1;
end
clear i;
```

```
for i=1:size
    for j=1:469
        pp=spline(x(i,:),y(i,j,:));
        t=ppval(pp,xd);
        for k=1:51
            ys(i,j,k)=t(k);
        end
    end
end
f=ys;
end
```

8. 计算总平均误差成本（fw.m）

```
%计算总平均误差成本
function f=fw(size,ys,data)
w=zeros(1,size);
for S=1:size
    for i=1:469
        for j=1:51
            t=abs(ys(S,i,j)-data(i,j));
            if t<=0.5 continue;
            end
            if t<=1
                w(S)=w(S)+0.5;
                continue;
            end
            if t<=2
                w(S)=w(S)+1.5;
                continue;
            end
            if t<=3
                w(S)=w(S)+6;
                continue;
            end
            if t<=5
                w(S)=w(S)+12;
                continue;
            end
            w(S)=w(S)+25;
        end
    end
end
for i=1:size
    t(i)=w(i)/469;
end
```

```
f=t;
end
```

9. 计算总平均成本（fA.m）

```
%计算总平均成本
function f=fA(size,point,w)
A=zeros(1,size);
cost=12*point;                          %最初测定成本
for i=1:size
    A(i)=w(i)+cost;
end
f=A;
end
```

10.计算最小值的下标（findex.m）

```
%计算最小值的下标
function f=findex(size,A)
index=1;
for i=2:size
    if A(i)<A(index)
        index=i;
    end
end
f=index;
end
```

11.打印当前最优解(fprint.m)

```
%打印当前最优解
function fprint(A,index,point,xp,G)
fprintf('G=%d cost:%f\n',G,A(index));
for j=1:point
    fprintf('%d ',xp(index,j));
end
fprintf('\n\n');
end
```

12.计算存活概率（flive.m）

```
%计算适应度分布区间
function f=flive(size,point,A,xp)
maxn=10000;
for i=1:size
    temp(i)=maxn/(A(i)^3);
end

Sum=0;
for i=1:size
    for j=1:i-1
        if temp(i)==temp(j)
```

```
            flag=1;                    %判断是否相等
            for k=1:point
                if xp(i,k)~=xp(j,k)  %若成本值相等，再检查取点方案是否相同
                    flag=0;
                    break;
                end
            end
            if flag==1
            temp(i)=0;                 %一旦重复，将后一点的概率置零
            break;
            end
        end
    end
    Sum=Sum+temp(i);
end

live(1)=temp(1)/Sum;
for i=2:size
    live(i)=temp(i)/Sum+live(i-1);
end;
f=live;
end
```

13.计算不同个体被选中的次数(fnum.m)

```
%计算不同个体被选中的次数
function f=fnum(size,live)
num=zeros(1,size);
for i=1:size
    r(i)=rand;
end
for i=1:size
    j=1;
    if r(i)<=live(1) num(1)=num(1)+1;
        continue;
    end
    while r(i)>live(j)
        j=j+1;
    end
    num(j)=num(j)+1;
end
f=num;
end
```

14.计算选择结果（fselect.m）

```
%计算选择结果（二进制表示）
function f=fselect(size,point,num,code)
```

```
s=zeros(size,point,6);
i=1;
for j=1:size
    for k=1:num(j)
        for p=1:point
            for q=1:6
                s(i,p,q)=code(j,p,q);
            end
        end
        i=i+1;
    end
end
f=s;
end
```

15.交叉（fcross.m）

```
%交叉运算(可能产生不在 1-51 之间的个体，先不作处理，当变异运算时再淘汰)
function f=fcross(size,point,select)
cross=zeros(size,point,6);
rd=randperm(size);                  %打乱 size 下标,随机交叉
for i=1:2:size
    for p=1:point
        r=randi(6);                 %随机设定交叉点位置
        for q=1:r
            cross(i,p,q)=select(rd(i),p,q);
            cross(i+1,p,q)=select(rd(i+1),p,q);
        end
        for q=r+1:6
            cross(i,p,q)=select(rd(i+1),p,q);
            cross(i+1,p,q)=select(rd(i),p,q);
        end
    end
end
f=cross;
end
```

16.变异（fmuta.m）

```
%计算变异结果
function f=fmuta(size,point,cross)
for i=1:size
    for j=1:point                   %point 个点，每个点均可能变异
    prob=randi(50);                 %变异概率 1/(50*6)=1/300
        if prob==1                  %变异点固定为二进制的最后一位或两位
            a=4+randi(2);
            if cross(size,j,a)==1
            cross(size,j,a)=0;
```

```
            else
                cross(size,j,a)=1;
            end
        end
    end
end
f=cross;
end
```

17.将二进制转回十进制（fback.m）

```
%将二进制转换为十进制
function f=fback(size,point,muta,index,xp)
t=zeros(size,point);
for i=1:size
    for p=1:point
        for q=1:6
            t(i,p)=t(i,p)+muta(i,p,q)*(2^(6-q));
        end
    end
end

for i=1:size                %去除不符合要求的点(用 1-51 之间的随机数代替)
    for j=1:point
        if t(i,j)>51||t(i,j)<1
            t(i,j)=randi(51);
        end
    end
end

for i=1:size                %去重(用 1-51 之间的随机数代替)
    same=1;
    while(same==1)
        same=0;
        for j=1:point-1
            for k=j+1:point
                 if t(i,j)==t(i,k)
                    same=1;
                    t(i,j)=randi(51);
                 end
            end
        end
    end
end

for i=1:size                    %排序
```

```
    for j=1:point-1
        for k=j+1:point
            if t(i,k)<t(i,j)
                tmp=t(i,k); t(i,k)=t(i,j); t(i,j)=tmp;
            end
        end
    end
end

for i=1:size
    for j=1:point
        xp(i,j)=t(i,j);
    end
end
f=xp;
end
```