统计推断在数模转换系统中的应用

组号：03 小组成员：刘昊 5130309302　彭昊玥 5130309316

摘要：在数学物理以及工程科学中，实验始终扮演着重要的角色，而在实验中得到的数据往往具有数量大，随机性强的特点。随着数理统计学科的发展，通过建立数学模型来解决实际问题已成为很有效的方法和途径。本课题要求为某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。而监测模块中传感器部件的输入输出特性呈明显的非线性。为了了解实际的特性曲线信息、研究对象系统的输入输出函数关系。在工程实际中，若对每个样品都进行完整测定，则既费时又高成本。若能减少需要观测的数值组数量，则可以提高工效。针对这一矛盾，就必须通过一个优化采样的算法来解决。 而我们这一次主要采用的方法是

关键字：插值、拟合；一维插值，样条插值；启发式搜索，遗传算法

1 引言

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。

2 数学模型的建立

为了对本课题展开有效讨论，需建立一个数学模型，对问题的某些方面进行必要的描述和限定。

监测模块的组成框图如图 1。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）。

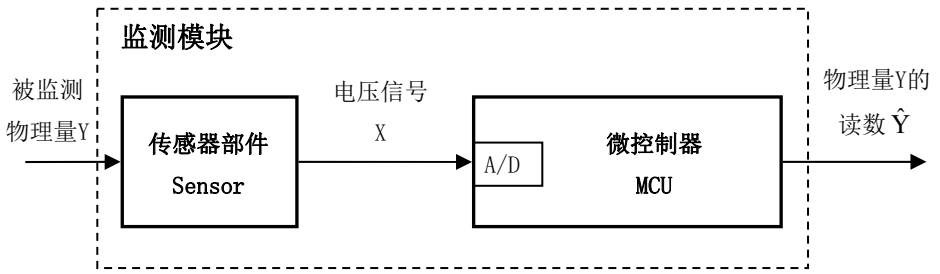

图 1 监测模块组成框图

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。

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， $i = 1, 2, 3, ..., 50, 51$。

2.1 传感部件特性

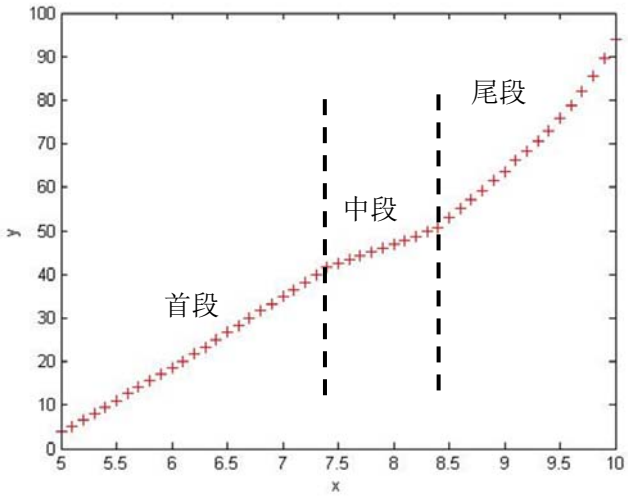

图 2 传感特性图示

一个传感部件个体的输入输出特性大致如图 2 所示，有以下主要特征：

- 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。

为进一步说明情况，图 3 对比展示了四个不同样品个体的特性曲线图示。

2.2 标准样本数据库

前期已经通过试验性小批量生产，制造了一批传感部件样品，并通过实验测定了每个样品的特性数值。这可以作为本课题的统计学研究样本。数据被绘制成表格，称为本课题的“标准样本数据库”。

该表格以 CSV 格式制作为电子文件。表格中奇数行存放的取值，偶数行存放对应的取值。第 2i－1 行存放第 i 个样本的 X 数值，第 2i 行相应列存放对应的实测 Y 数值。

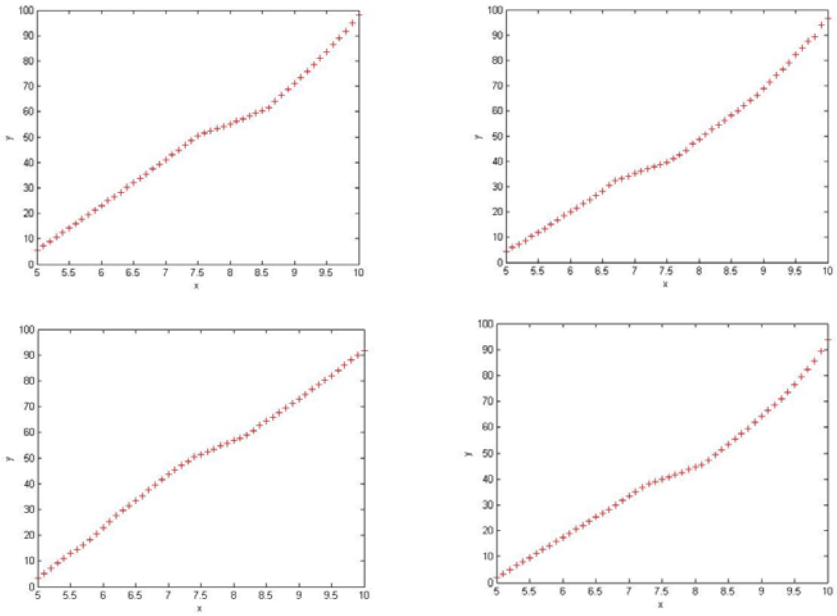

图 3 四个不同样本个体特性图示对比

2.3 成本计算

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

$$s_{i,j}=\begin{cases}0 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|\le 0.5\\ 0.5 & \text{if } 0.5<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 1\\ 1.5 & \text{if } 1<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 2\\ 6 & \text{if } 2<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 3\\ 12 & \text{if } 3<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 5\\ 25 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|>5\end{cases} \quad (1)$$

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。

- 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

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

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。

- 校准方案总体成本

按式（3）计算评估校准方案的总体成本，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。

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

总体成本较低的校准方案，认定为较优方案。

3 数据的拟合

从以上对于数据的初步简单分析中，我们可以看出：在本课题中，被监测的物理量 Y 与输出电压 X 并不具有线性关系，而且不同组数据之间，数据特性的一致性并不强，所以，需要寻找一种拟合方式对数据进行拟合，对于本例，我们选取了三次样条插值方法进行了尝试并对其进行实现。

3.1 样条插值

一维数据插值可以得到函数 y=f(x)，在进行插值时，随着数据点数目的增多以及数据点之间距离缩短，插值会变得越来越精确。但在数据量比较有限的情况下，通过合理地选择插值方法，可以得到比较满意的插值结果。

样条插值方法的主要原理和思想是，根据一组已知的数据点，找到一组拟合多项式进行拟合。在多项式拟合的过程中，保证每组相邻的样本数据，采用三次多项式拟合样本数据点之间的曲线。为了保证拟合结果的唯一性，在这个三次多项式的采样处使用一阶、二阶导数加以约束。因此，所有的样本点之间的数据也都保证满足一阶、二阶导数连续。

3.2 插值的过程

在本例中，我们选取 6 个采样点进行三次样条插值，其中首尾两个采样点选定，另外四个在第 2~49 点中随机生成。选取采样点之后，通过 MATLAB 中的 spline 函数对其进行三次样条插值，拟合出相应函数，然后将样本中各点及其所对应的观测值代入拟合函数中，通过函数值与观测值的比较得出样本所对应的成本。（MATLAB 代码在附录中给出）

4 遗传算法的简介与实现

在本例中，我们在选择测定点的组合方案时，选取的启发式搜索算法为遗传算法（Genetic Algorithm. GA）。

4.1 遗传算法简介

遗传算法是模仿自然界生物进化机制发展起来的随机全局搜索和优化方法，它借鉴了达尔文的进化论和孟德尔的遗传学说。其本质是一种高效、并行、全局搜索的方法，他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。

遗传算法有三个基本操作：选择、交叉和变异。基本遗传算法是一种群体型操作，该操作以群体中的所有个体为对象，只使用基本遗传算子：选择算子、交叉算子和变异算子，其遗传进化操作过程简单，容易理解，是其他一些遗传算法的基础，它不仅给各种遗传算法提供了一个基本框架，同时也具有一定的应用价值。选择算子、交叉算子和变异算子是遗传算法的 3 个主要操作算子，它们构成了所谓的遗传操作，使遗传算法具有了其他传统方法没有的优点。

下面简要介绍三种基本遗传算子：

（1）选择运算使用比例选择算子。比例选择因子是利用比例于各个个体适应度的概率决定其子孙的遗传可能性。若设种群数为 M，个体 i 的适应度为 f_i，则个体 i 被选取的概率为

$$P_i=f_i/\Sigma f_k$$

当个体选择的概率给定后，产生[0,1]之间的均匀随机数来决定哪个个体参加交配。若个体的选择概率大，则能被多次选中，它的遗传基因就会在种群中扩大；若个体的选择概率小，则被淘汰。

（2）交叉运算使用单点交叉算子。只有一个交叉点位置，任意挑选经过选择操作后种群中两个个体作为交叉对象，随机产生一个交叉点位置，两个个体在交叉点位置互换部分基因码，形成两个子个体。

（3）变异运算使用基本位变异算子或均匀变异算子。为了避免问题过早收敛，对于二进制的基因码组成的个体种群，实现基因码的小概率翻转，即 0 变为 1,1 变为 0。

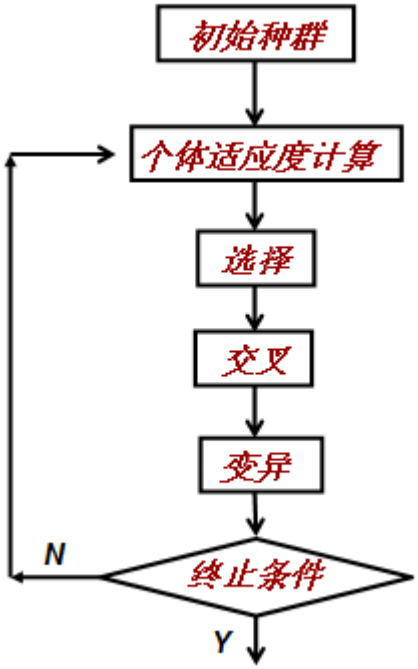

图 4 遗传算法流程图

自然遗传学	人工遗传算法
染色体	解的编码（数据、数组、位串）
基因	解中每一分量的特征（特性、个性、探测器位）
等位基因	特性值
基因座	串中位置
基因型	结构
表现型	参数集、解码结构、候选解
遗传隐匿	非线性
个体	解

适者生存	在算法停止时，最优目标值的解有最大的可能被留住
适应性	适应度函数值
群体	选定的一组解（其中解的个数为群体的规模）
复制	根据适应函数值选取的一组解
交配	通过交配原则产生一组新解的过程
变异	编码的某一个分量发生变化的过程

表 1 遗传学和遗传算法中基本用语对照表

4.2 遗传算法在本例中的应用

在本例中，我们在进行模拟时，选取的种群数为 100，群体大小为 51，规定在遗传时的交叉概率为 0.5%，变异概率为 0.01%，去种群成本的倒数作为适应度，经过 500 代循环，选取成本值最低的一组解作为最佳的取点方案。

4.2.1 选择

选择过程中，基于之前实现的种群成本计算函数 avgcost，取倒数作为染色体的适应度。然后对所有染色体的适应度求和，取每条染色体的适应度与适应度总和的比值作为该染色体的选择概率，将个选择概率一次累加后得到[0,1]内递增的累计概率，通过在[0,1]内生成随机数得到子代染色体。

4.2.2 交叉

交叉之前，通过之前规定的交叉概率求得参与交叉的染色体条数，再由随机数生成需要参加交叉的染色体以及交叉点的位置，通过将两条染色体交叉点后的基因互换，得到两条新染色体。

4.2.3 变异

变异过程的实现相对简单，对于每个基因，通过生成的[0,1]之间的随机数与变异概率的比较决定其是否变异，若变异则突变成对应的等位基因，即若原基因为 0，则变异为 1；若原基因为 1，则变异为 0。

4.2.4 结果

本次遗传算法模拟了 500 代的交配过程，每一代均需运行半分钟以上的时间，计算量较为庞大。在 100 代之后，结果开始收敛，最终结果在 96~97 左右。最近一次的运行结果及其对应的个体为

Generation10;

98.4755

1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0

1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1

Generation30;

97.3401

1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0

1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1

Generation100

96.8785

1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0

1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1

Generation200

96.8786

1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0

1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1

Generation500

96.8787

1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0

1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1

5 结论

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。为了更好地提高效率，我们选择 7 个“事先观测点”对曲线进行拟合，并得到了在评价函数下得分为 96.8787 分的 1 组“事先观测点”。它们的对应组数为【1,10,20,28,33,45,51】,计算时间为 30min。在该课题中我们采用了遗传算法，其产生的结果和初始种群数以及遗传的代数有很大关系。而由于本组成员对 MATLAB 属于初学状态，不太会将循环变为矩阵运算，所以导致程序的运算时间很长，由于“事先观测点”个数的限制，可能得到的最优解并不是所有数据的最优解，但是基本实现了课题要求，完成了课题任务。

参考文献：

[1] 周建兴、岂兴明、矫津毅、张延伟等 MATLAB 从入门到精通.北京:人民邮电出版社,2012,195~203.

[2] 雷英杰、张善文 MATLAB 遗传算法工具箱及应用.西安:西安电子科技大学出版社,2014,1~22.

附录：
程序代码：

```
function f=geneticalgorithm()
popsize=51; %群体大小
N=100;%种群数
pc=0.5; %交叉概率
generation=500;%循环代数
pm=0.01; %变异概率
A=textread('C:\Users\phy\Desktop\20141010dataform.txt');
group=zeros(N,popsize);%种群矩阵(行表示种群，列表示群体）
groupfitness=zeros(1,N);%种群适应度矩阵
bestvalue=852;
best=zeros(1,popsize);

for i=1:N
    group(i,1)=1;group(i,popsize)=1;
    g=round(rand(1,4)*48)+2;
    for j=1:4
        group(i,g(j))=1;
    end
end
cost=zeros(1,N);
for i=1:N
    cost(i)=avgcost(A,group(i,:));
end

[value,id]=min(cost);
bestvalue=value;
best=group(id,:);

for g=1:generation%循环代数
    for k=1:N%计算适应度
        groupfitness(k)=1/avgcost(A,group(k,:));
    end
    sumfitness=sum(groupfitness(:));
    fitrate=zeros(1,N);
    for t=1:N
        fitrate(t)=groupfitness(t)/sumfitness;
    end
    groupafterchoose=zeros(N,popsize);
    fitrate=cumsum(fitrate);
    for i=1:N
        a=rand(1,1);
        if(a<=fitrate(1))
```

```
            groupafterchoose(i,:)=group(1,:);
        end
        for j=1:N-1
            if (fitrate(j)<a&&a<=fitrate(j+1))
                groupafterchoose(i,:)=group(j,:);
            end
        end
    end
    groupafterchoose(1,:)=best;
    %按照 pc 决定参与交叉的染色体数，从经过选择后的数组中选择染色体配对交叉
    for i=1:pc*N
    cut=randi(popsize-1);%交换的位置随机选择
    change1=randi(N);%随机选取两个个体进行交叉
    change2=randi(N);
    tmp=groupafterchoose(change1,cut+1:popsize);%
    groupafterchoose(change1,cut+1:popsize)=groupafterchoose(change2,cut+1:popsize);
    groupafterchoose(change2,cut+1:popsize)=tmp;
    end
    %选择出交叉后的最优个体，并与先前的最优个体比较
cost=zeros(1,N);
for i=1:N
    cost(i)=avgcost(A,groupafterchoose(i,:));
end
[value,id]=min(cost);

if value>bestvalue
    groupafterchoose(1,:)=best;
end

if value<=bestvalue
    bestvalue=value;
    best=groupafterchoose(id,:);
end

groupaftercross=groupafterchoose;
%每一个种群的每一个个体以 pm 的概率变异
for i=1:N%变异
        for j=2:popsize-1
            tmp=unifrnd(0,1);%0~1 随机数
            if(tmp<=pm&&groupaftercross(i,j)==1)
                groupaftercross(i,j)=0;
            end
            if(tmp<=pm&&groupaftercross(i,j)==0)
                groupaftercross(i,j)=1;
```

```
                end
            end
end
%选择出变异后的最优个体，并与先前的最优个体比较
cost=zeros(1,N);
for i=1:N
    cost(i)=avgcost(A,groupaftercross(i,:));
end
[value,id]=min(cost);

if value>bestvalue
    groupaftercross(1,:)=best;
end

if value<=bestvalue
    bestvalue=value;
    best=groupaftercross(id,:);
end

group=groupaftercross;
%展示
disp(['Generation',num2str(g),':']);
disp([num2str(bestvalue),'          ',num2str(best)]);
end
end
function y = avgcost(A,array)
start=1;final=51;
sum=0;
for i=1:469
    x=A(2*i-1,start:final);
    y=A(2*i,start:final);
    sum=sum+cost(x,y,array);
end
y=sum/469;
end
function f= cost(x,y,array)
j=0;everycost=0;num=51;
choosex=zeros(1,sum(array));
choosey=zeros(1,sum(array));
for i=1:num
    if(array(i)==1)
        j=j+1;
        choosex(j)=x(i);
        choosey(j)=y(i);
```

```
        end
end
x1=5.0:0.1:10.0;
        y1=interp1(choosex,choosey,x1,'spline');

d=zeros(1,num);
for i=1:num
        d(i)=abs((y(i)-y1(i)));
end
for i=1:num
        if(d(i)<=0.5)
                everycost=everycost+0;
        end
        if(d(i)>0.5&&d(i)<=1)
                everycost=everycost+0.5;
        end
        if(d(i)>1&&d(i)<=2)
                everycost=everycost+1.5;
        end
        if(d(i)>2 && d(i)<=3)
                everycost=everycost+6;
        end
        if(d(i)>3 && d(i)<=5)
                everycost=everycost+12;
        end
        if(d(i)>5)
                everycost=everycost+25;
        end
end
f=everycost+12*sum(array);
end
```