统计推断在数模转换系统中的应用

作者：高天昊 5140309079

李瑞泽 5140309159

组别：67 组

日期：2015 年 12 月 1 日

摘要：对于某种装置系统，输入和输出存在一定的受控关系，但是这种受控关系是非线性的并且个体差异较大。因此在大批量生产中，需要考虑到该受控关系。 在工程实际中，若对每个样品都进行数值的完整测定，则既费时又消耗成本。本课程在以某电子产品内部传感传感器部件的输入输出特性为研究对象，在确保测量 精度的前提下，运用统计方法结合 Matlab 四百多组样本中电压信号 X 和物理量 Y 的数据关系进行分析，对样本数据进行拟合，运用遗传算法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。
关键词：统计推断，传感特性，拟合，遗传算法，成本，Matlab

Application of Statistical Inference in DA Inverting System

ABSTRACT:

For a unit system, there are certain controlled relationship between input and output, but this kind of controlled relationship is nonlinear and individual differences. So in mass production, the need to consider the relationship between the controlled. In engineering practice, if the value of each sample complete measurement, is both time consuming and cost.The course in order to input and output characteristics of the internal sensor components of some electronic product as the research object,on the premise of ensuring accuracy, were analyzed by using statistical method in combination with Matlab more than 400 samples the voltage signals X and Y data between physical quantities, by fitting the sample data, and use the genetic algorithm to design a sensor characteristics calibration the reasonable cost for the module of batch production (calibration procedure) scheme.
Key words: Statistic Interference, Sensing characteristics, Fit, Genetic Algorithm, Cost, Matlab

1 引言

本次试验中，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。检测的工序一方面要保证精度，另一方面要考虑到控制操作成本，因此我们需要研究该系统的特性来找出一种方法衡量产品的品质。本次研究中，我们运用遗传算法，结合拟合方法对近五百组数据进行分析，结合 Matlab 语言，找出成本相对最小的传感特性校准（定标工序）方案。这是一个组合组合优化问题，但备选的组合方案数量巨大，是典型的 NP-hard 问题。

2 系统的简单描述与数学模型

这个问题抽象为一个数学问题就是在一个部件特性曲线未知的前提下，求取 n 个电压值进行测量，得出 7 个样本数据点，通过这 n 个点拟合出部件的特性曲线且测量成本与误差成本之和要尽量的小。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。

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 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 $y\hat{} = f(x)$ 的过程，其中 x 是 X 的取值， $y\hat{}$ 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

$X \in \{x_1, x_2, x_3, \ldots, x_{50}, x_{51}\} = \{5.0, 5.1, 5.2, \ldots, 9.9, 10.0\}$。相应的 Y 估测值记为 $\hat{y}_i = f(x_i)$， Y 实测值记为 y_i， $i = 1, 2, 3, \ldots, 50, 51$。

成本计算函数函数：

单点定标误差成本：i 表示样本序号，j 表示观测点序号

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

$$S_i = \sum_{j=1}^{51} s_{i,j} + 12N_i$$

对某一个样品 i 的定标成本 ，Ni 为该样品定标时所测定的点数。

第一项：误差成本 第二项：测定成本

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

定标方案总成本:

注：1.基于《标准样本库数据》，我们求取统计平均意义上的最好方法。

2.总成本最低的方案，对特定样品而言不一定是最低成本。

3 方案选择

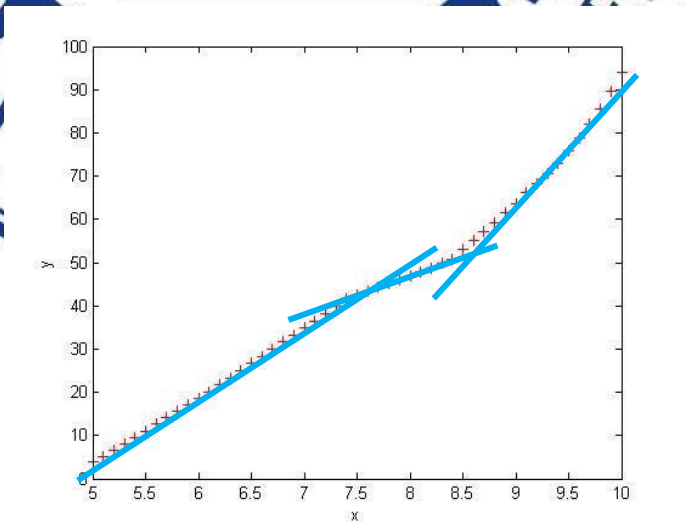

图 3-1

X-Y 数据大致分布曲线通过对原始数据的查看与分析，发现每组数据的 X-Y 曲线有这很大的相似性，大致可以看成由三段可近似看为直线段的曲线组成。曲线特性如图 3-1 所示。

由曲线的分布特性，我们联想到了三次函数，三次函数的曲线特性就大致分为三段走，于是我们首先采用的就是三次多项式的拟合这一方案。

3.1 拟合函数的选取

由于取点函数的多样性，我们并不知道针对课题的数据，选用何种拟合函数，可得成本最优，于是我们选取了两个拟合函数进行对比和探索。

3.1.1 三次多项式的拟合

由于前面已经讨论到，取点个数越多，则拟合曲线越为精准，而取点次数会影响成本总值，取点次数越多，成本越高，所以我们在此要寻找一个最优的取点方案，首先要找到一个最优的取点个数 n，能让总成本达到最低。我们编写了相关三次多项式拟合的可以更改取点数目 n 的 matlab 程序，通过 1000 次的穷举，我们可以大致地认为在这个取点数目 n 下，已经找到最优的取点方案了，通过对不同取点数目 n 下的最优成本的分析，我们可以寻找得到最优的取点数目 n。

3.1.2 三次样条插值的拟合

通过对数据曲线的进一步分析，以及了解到了三次样条插值法这一拟合方案，我们预测，由于曲线三段较为线性，那么通过三次样条插值法得出的拟合曲线也许更为优质。

首先先来介绍一下三次样条插值法，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。定义:函数 $S(x)\in C2[a,b]$ ，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。

我们不必刻意去追究三次插值法的实现，因为 matlab 程序已经为我们很好地将这一插值函数实现。

于是，和三次多项式最优拟合方案探索思想类似，我们同样也通过编写可以更改取点数目 n 的三次样条插值的拟合函数的 matlab 程序来寻求最优的 n 取值。

3.2 算法的选择

3.2.1 暴力穷举

若用暴力穷举法，解决以上问题需要尝试约 1.54 亿种不同组合，显然这是不切实际的

3.2.2 遗传算法

1.思想概述

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。 所以在此问题中选择该方法。

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。

3.2.2.1 遗传算法介绍

遗传算法是具有“生成+检 测 ”的迭代过程的搜索算 法。基本流程如图 3-2 所示。可见,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。选择(selection)、交叉(crossover)和变异(mutation) 是 遗 传 算 法 的 三个主要操作算子。遗传算法包含如下 6 个基本要素:

(1)参数编码:由于遗传算法不能直接处理解空 间的解数据,因此必须通过编码将它们表示成遗传空 间的基因型串结构数据。

(2)生成初始群体:由于遗传算法的群体型操作 需要,所以必须为遗传操作准备一个由若干初始解组 成的初始群体。初始群体的每个个体都是通过随机方 法产生的。

(3)适 应 度 评 估 检 测 :遗 传 算 法 在 搜 索 进 化 过 程 中一 般 不 需 要 其 他 外 部 信 息 ,仅 用 适 应 度(fitness)值来评估个体或解的优劣,并作为以后遗传操作的依 据。

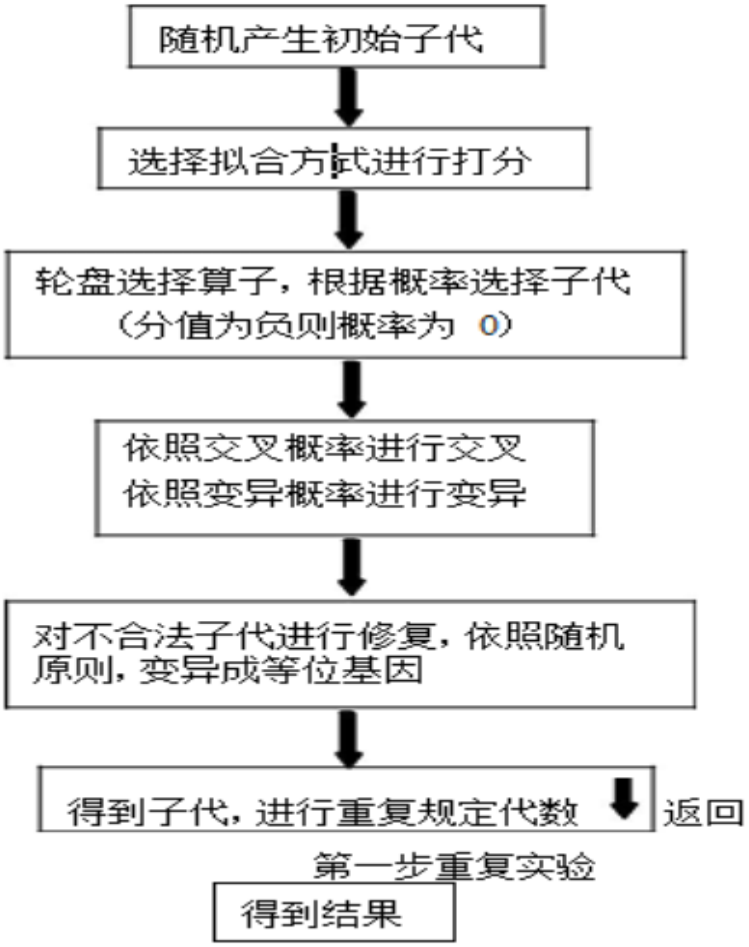

图 3-2 遗传算法流程图

(4) 选择(selection)：选择或复制操作是为了从当前群体中选出优良的个体，使它们有机会作为父代为下一代繁殖子孙。个体适应度越高，其被选择的机会就越多。本文采用与适应度成比例的概率方法进行选择。具体地说，就是首先计算群体中所有个体适应度的总和，再计算每个个体的适应度所占的比例，并以此作为相应的选择概率。

(5) 交叉操作：交叉操作是遗传算法中最主要的遗传操作。简单的交叉(即一点交叉)可分两步进行：首先对种群中个体进行随机配对，其次，在配对个体中随机设定交叉处，配对个体彼此交换部分信息。

(6)变异：变异操作是按位(bit)进行的，即把某一位的内容进行变异。变异操作同样也是随机进行的。一般而言，变异概率 p_m 都取得较小。变异操作是十分微妙的遗传操作，它需要和交叉操作配合使用，目的是挖掘群体中个体的多样性，克服有可能限于局部解的弊病。[2]

遗传算法就是模拟进化论的优胜劣汰的规则，并加有概率性质的随机遗传的算法。初始群体的每个个体都是通过随机方法产生的。仅用适应度值来评估个体或解的优劣，并作为以后遗传操作的依据。选择或复制操作是为了从当前群体中选出优良的个体，使它们有机会作为父代为下一代繁殖子孙。个体适应度越高，其被选择的机会就越多。交叉操作是遗传算法中最主要的遗传操作。简单的交叉(即一点交叉)可分两步进行：首先对种群中个体进行随机配对；其次，在配对个体中随机设定交叉处，配对个体彼此交换部分信息。变异操作是指每个父代个体都有一定的概率发生随机的基因突变。评价基因好坏的依据就是适应度。在遗传算法的步骤中起着突出作用的其实是变异。正因为有了变异，算法才存在在全局范围内寻找到最优解的可能性。但是遗传算法需要通过反复试验，才有可能给出最优解。

3.2.2.2 遗传算法思想在三次样条插值的应用

在先前的探索过程中，我们从结果中可以看出使用三次样条插值法并且取点数为 6 进行拟合时，可取得较为优秀的取点组合，故我们首先依据该结论将遗传算法仅用于取点数为 6 的三次样条插值法。我们选取的种群个体数为 50，遗传选择 100 代。首先从所给的数据中随机选择 50 个个体，取定第一个点和最后一个点，然后随机选择四个点，构成一条染色体，然后对 50 条染色体分别计算成本，以成本的倒数作为每一个个体的适应度，再对种群进行初始化。初始化的过程类似于赌轮算法，即对之前选择的 50 个个体，分别计算出他们的适应度所占的比例，即适应比例。这样，50 个个体按比例分布在赌轮上，转动赌轮 50 次，即可得到 50 个个体，这 50 个个体就是初始化之后的种群。在这个种群中，适应比例高的个体出现的次数多，适应比例低的个体出现的次数少，某些适应比例过低的个体不会出现在种群中，这样相当于对种群中的个体进行了选择，以此为初始种群，使每一个个体等比例自由交配即可。对交流之后获得的新种群，继续运用赌轮算法淘汰，然后检测。如果结果优化，则继续交配，如果结果没有优化，则此次交配作废，继续沿用之前的种群来进行基因交流。

在这个条件下，通过运行程序，我们得到的最优的取点方案以及成本为：

1　　10　　21　　31　　43　　51

97.3827

与先前穷举得出的最优解相比，有了一定程度的优化。

3.2.2.3 基于全局的遗传算法的应用

先前的算法只是大致应用了遗传算法的思想，而并未完全的应用遗传算法进行。我们只是通过穷举大致的了解了成本关于 n 的分布情况，然后基于这一结论进行的探索。真正的遗传算法的应是随机产生取点数 n 进行函数拟合的算法，而且我们只是大致以为当取点数目为 6 时，成本最优，不排除成本最优解并不是取点数为 6 的可能。所以基于这一想法，我们对不同的取点方案进行过测试，测试过后发现，当取点数目为 6 时，成本比较低。测试并非穷举，只是对几个不同的数值进行测试，结果可能并不正确，但有一定的依据。

3.3 结果的分析

3.3.1 拟合函数选择结果的分析

就拟合函数的选取，我们认为，三次样条插值法有它的合理性。首先根据数据的分布情况，已经在上文中有过介绍，就是数据曲线大致分为前段，中段，后段这三段，并且这三段的线性度较高，正好符合三次插值法是取两点之间连线以进行拟合的特性，所以就这个课题而言，还是三次样条插值法精确度更高。

由于取点方案是通过对不同的方案进行测试而得到的，有较大的随机性，理论上的依据可能比较弱，但还是有一定的参考价值的。

3.3.2 遗传算法的分析

在选定了三次样条插值法之后，我们就要通过对取点的优化来更好地进行函数拟合以使成本最小。我们首先通过不同取点数目 n 值下的测试来探索最优的取点数目，最终得出结论是当取点数目为 6 时，成本最优。我们在此结论基础上，运用遗传算法思想编写程序，通过 100 代的循环，最后得出了最低成本为 97.3827 的取点方案。

由于取点方案是通过对不同的方案进行测试而得到的，有较大的随机性，理论上的依据可能比较弱，但还是有一定的参考价值的。

(函数代码附在附页)

（附页）

遗传算法

```
function final_result = geneticalgorithm(A)

generation=100; %遗传代数

popsize = 50; %种群中个体数量

best = zeros(1,6); %经测试，采用六个点进行定标

pop = zeros(popsize,6); %一个矩阵，每一行代表一种取点方案，即一条染色体

popfitness = zeros(1,popsize); %每一条染色体对应的适应度（是其平均成本的倒数）

fitrate = zeros(1,popsize); %每种适应度的比例

cost = zeros(1,popsize); %每一条染色体对应的定标成本

bestValue = 0;

%初始化种群

for i=1:popsize
    pop(i,1)=1; pop(i,6)=51;
    p = randperm(49)+1;
    pop(i,2:5) = p(1:4);
end

for i=1:popsize    %计算每一个个体的成本
    cost(i) = test_ur_answer(A,pop(i,:));
end

[value,index] = min(cost); %找出最优成本

bestValue = value;
```

```
best = pop(index,:);


for i=1:popsize %计算适应度（以成本倒数作为适应度）
    popfitness(i) = 1/cost(i);
end
sumpopfitness = sum(popfitness);
for i=1:popsize    %计算适应度所占的比例
    fitrate(i) = popfitness(i)/sumpopfitness;
end



pop_tmp = zeros(popsize,6);
fitrate = cumsum(fitrate);
for i=1:popsize    %为种群赋值，按照适应度所占的比例，比例越高在种群里的数量就越多
    a = rand(1,1);    %某些适应度较低的个体将不会出现在种群里
    if a<fitrate(1)
        pop_tmp(i,:) = pop(1,:);
    end
    for j=1:popsize-1
        if a<fitrate(j+1) && a>=fitrate(j)
            pop_tmp(i,:) = pop(j+1,:);
        end
    end
end
pop_tmp(1,:) = best;
pop = pop_tmp;



for i=1:generation %进行循环
    for j=1:generation/2 %基因交流，染色体交换
        while 1
            pos = round(5*rand())+1;
            chromosome1 = randi(popsize);
            chromosome2 = randi(popsize);
            tmp = pop_tmp(chromosome1,1:pos);
            pop_tmp(chromosome1,1:pos) = pop_tmp(chromosome2,1:pos);
            pop_tmp(chromosome2,1:pos) = tmp;
            x1 = pop_tmp(chromosome1,:);
```

```
        x2 = pop_tmp(chromosome2,:);
        if length(x1)-length(unique(x1))==0 && length(x2)-length(unique(x2))==0
            break; %这个 if 语句保证染色体序列中不会出现相同的点
        end
    end

end

for k=1:popsize %算子代的成本
    cost(k) = test_ur_answer(A,pop_tmp(k,:));
end

[value,index] = min(cost);

if value < bestValue %如果结果优化
    bestValue = value;
    best = pop_tmp(index,:);

    for k=1:popsize %算新种群的适应度
        popfitness(k) = 1/cost(k);
    end
    sumpopfitness = sum(popfitness);
    for k=1:popsize
        fitrate(k) = popfitness(k)/sumpopfitness;
    end
    fitrate = cumsum(fitrate);

    for k=1:popsize %新种群进行淘汰
        a = rand(1,1);
        if a<fitrate(1)
            pop(k,:) = pop_tmp(1,:);
        end
        for t=1:popsize-1
            if a>=fitrate(t) && a<fitrate(t+1)
                pop(k,:) = pop_tmp(t+1,:);
            end
        end
    end
```

```
            pop(1,:) = best;
            pop_tmp = pop;

        else
            pop_tmp = pop;
        end
end


best
final_result = bestValue;
```

成本函数

```
function ave_num = test_ur_answer(A,array)

my_answer_n=size(array,2);


[M,N]=size(A);
nsample=M/2; npoint=N;
x=zeros(nsample,npoint);
y0=zeros(nsample,npoint);
y1=zeros(nsample,npoint);
for i=1:nsample
    x(i,:)=A(2*i-1,:);
    y0(i,:)=A(2*i,:);
end
my_answer_gene=zeros(1,npoint);
my_answer_gene(array)=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

    % 请把你的定标计算方法写入函数 mycurvefitting

    y1(j,:)=mycurvefitting(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;


ave_num = cost;
```

三次插值

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

x=[5.0:0.1:10.0];

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

end
```