统计推断在数模与模数转换中的应用

组号：25 姓名：冯哲彬 学号：5130309357 姓名：张文博 学号：5130309359

摘要：本文为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。运用数学软件 Matlab，实现通过遗传算法探索基于事前观测点的拟合方法，从而得到待测电子产品的系统的输入输出关系，用以方便对之后生产的电子产品特性的测定。

关键词：统计推断，遗传算法，Matlab

1. 引言：

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。现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 测量对象的分析

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）。

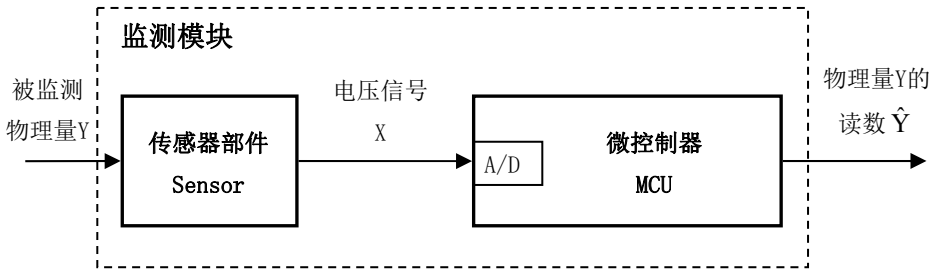

图 1 监测模块组成框图

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。

1.2 数据处理的方法

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， $i = 1,2,3,...,50,51$。

由于即使假设共选取 6 个点，通过穷举法需要测试取点的组合方案也有 $C_{51}^{6} = 18009460$ 种，加之未必选取 6 个点，那么工程量将会进一步扩大。故将主要运用遗传算法来进行主要取点，拟合，最终测定成本的过程。

2. 基于遗传算法的实现

2.1 遗传算法概述

关于遗传算法，我们的理解是一种模拟生物进化过程而衍生出的一种启发式搜索算法，用以在无穷大、不可解问题中（NP-hrad 问题），寻找出最优解的一种算法。主要过程为

（1）随机选择适当的种群规模。在此课题中便是将随机选点的过程。

（2）进行逐代衍化。共有一下可能的情况，一是直接由父代传输给子代不改变任何基因；二是交叉遗传，即两个父代交叉基因而生成的新的子代；三是变异，即基因的随机改变而生成新的子代。评价基因好坏的依据就是适应度。

在遗传算法的步骤中变异起着突出作用。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（一开始的种群可能不存在最优的基因）。但是遗传算法需要通过反复试验，才有可能给出最优解。

（3）产生出代表新的解集的种群。这个过程将导致种群像自然进化一样的后生代种群比前代更加适应于环境，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。具体实现中需要足够多代的衍化才能找到最优的个体。

2.2 适应度函数

根据《课程设计课题和要求》一文，通过前期试验性小批量生产，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，我们可以知道 X-Y 特性图大致为图 2 所示，差距主要为中段起始和结束的位置。

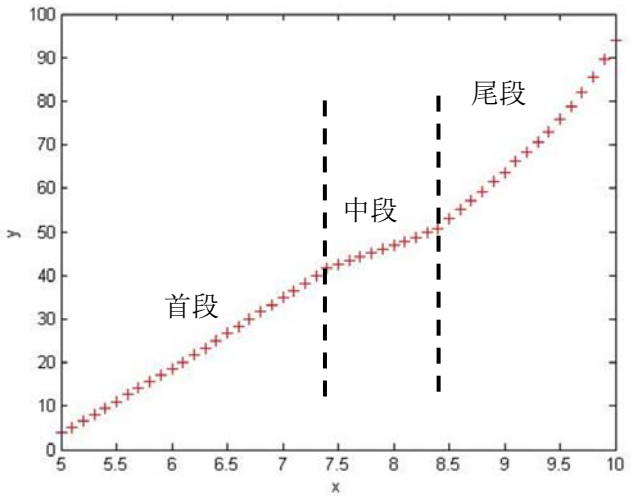

图 2　　X-Y 特性图

考虑到数据曲线为首段与尾端斜率较大而中段斜率较小，我们首先运用最为熟悉的三次多项式拟合的方法。我们对于课题的解题思路为，先随机选若干点，拟合出三次曲线：$F=AX^3+BX^2+CX+D$。再运用拟合出的曲线由选定点的 X 计算出 Y’值，即 $Y_{i,j}’=F(X_{i,j})$，并进行成本计算看所采取的选点策略是否最优。

对于 MATLAB 来说，三次多项式拟合十分简单在，运用 polyfit 函数即可。

2.3 成本计算与定标

课题主要要求在选点与误差中寻找最低的成本函数。在此我们需要规定一定的选点数目，及拟合方程来定标以确定成本。

$$s_{i,j}=\begin{cases}0 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|\le 1\\ 1 & \text{if } 1<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 2\\ 2 & \text{if } 2<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 3\\ 4 & \text{if } 3<\left|\hat{y}_{i,j}-y_{i,j}\right|\le 5\\ 10 & \text{if } \left|\hat{y}_{i,j}-y_{i,j}\right|>5\end{cases} \qquad (1)$$

由此运用单点定标误差成本式（1）进行初次的成本计算某一样本个体的定标成本（如式 2 所示,其中 q 为实行一次单点测定的成本，在本次实验中记为 20； n_i 为对样本 i 个体定标过程中的单点测定次数）

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

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。总体成本较低的校准方案，认定为较优方案。

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

而后将我们所随机选择出的若干点进行选择，交叉，变异过程的计算，每经过一代衍化，重新进行一次拟合曲线过程，得到三次曲线：$Fi=AnX_i^3+BnX_i^2+CnX_i+Dn$，并再次由此计算出由拟合曲线对应出的 Y’值，再次进行总体成本测定。最终检查进行 n 代迭代后的子代是否为其最优解，即总体成本函数所得出的值最低。

关于点的个数，显然，当点的个数上升时，$q\cdot n_i$ 项成本将会上升，而 $\sum_{j=1}^{51}s_{i,j}$ 项成本将会下降；同样，点的个数下降时，$q\cdot n_i$ 项将会下降，$\sum_{j=1}^{51}s_{i,j}$ 项成本将会上升。故在选点的过程中，极小值将会为最小值。

根据理论的推导，虽然每一次遗传所得到的选点可能不为最优解，甚至可能会得到比遗传之前更差的结果，但理应从总体角度去看，遗传算法应有一个曲折，并随着遗传代数的增加而有一个趋近于最优解的趋势。而我们每一次遗传都进行一次拟合便可以进行观察和验证这一点。

2.4 具体实现的过程

事实上，有些生物的繁殖是通过自我分裂实现的，即实行的是在 2.1（2）中所述的情况一和三以进行衍化。当然这类生物的变异将会是主要改变基因以最优的方式。虽然自然界

中变异的几率比较低，需要很多代衍化的才会发生，但我们可以加大变异的几率与变异的程度来加速这一过程。此即为我们的思想实现方法。

在 Matlab 的具体实现过程中，根据在课堂演示的过程中老师的建议，选点应为 6 个左右，故我们分别对选取 5、6、7 个点进行测试，下述过程以选取 6 个点为例。

一开始我们将运用随机数程序选取 10 组 6 个点作为 10 个种群，并以此进行拟合，算出适应度函数。然后进行变异，其方式为和改点值变异为改值周边的值（即 2 变异为 1 或 3 等），变异的点以及方式将以一定的随机方式而定。变异完成后，计算该代的 10 个种群的适应度函数，选取适应度最高的 10 个不同的种群作为新的种群，在进行下一代的衍化。在一代代的衍化之后，将会出现一个几乎不变的最优解。实现过程如图 3 所示（终止条件依老师建议设为 30 代）。

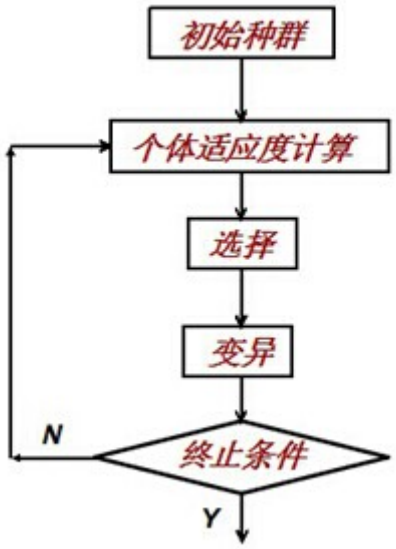

图 3 本实验中遗传算法实现过程图

3. 结论

根据对实验数据的统计分析以及 Matlab 的编程拟合实现，可得一下结论

（1）对于点的个数选取最终定为 6 个为宜，其最终得到的适应度最好，选定的点为[3,11,21,30,40,49],cost（总体成本）约为 93，即通过该 6 个点的测量，就能既节约成本又能得到较好的测量对象的特性。

（2）在实验过程中，我们发现当进行到第 12 代左右时，种群的个体分布就趋于稳定了。若读者想要已附录 1 的代码进行实验时可借以参考（同时注意运行时间较长）。

（3）同时我们也进行了对三次样条的测试，选点以 6 个点，发现总体成本值高于三次函数。由于数据未加以记录，在此将不予给出。

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.

附录 1：

关于遗传算法代码如下：

1.fitness.m

```
function fitvalue = Fitness(population,popsize)
for i=1:1:popsize
    fitvalue(i,1)=cost(population(i,:));
end
```

2.cost.m

```
function cost = cost(xx)

my_answer=xx;
my_answer_n=size(my_answer,2);

minput=dlmread('20141010dataform.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;

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
```

3.genetic.m

```
clc;
close all;
clear all;

Generationmax=30;

popsize=60;
point=6;
population=randpop(51,popsize,point);
generation=0;

while generation<Generationmax
    disp(generation);

population=selection(Fitness(population,popsize),population,popsize,point);
    population=repopulate(population,popsize,point);
    generation=generation+1;
end
disp(population(1,:));
```

4.gerand.m

```
function sample = gerand(mut,point)
flag=0;
list(1,1)=0;
list(1,point+2)=52;
list(1,2:point+1)=mut;
for i=2:1:point+1
    while flag==0
        x=randi(3);
        y=randi(2);
        if y==1&&list(1,i)+x-1<list(1,i+1)
            list(1,i)=list(1,i)+x-1;
            flag=1;
        end
        if y==2&&list(1,i)-x+1>list(1,i-1)
            list(1,i)=list(1,i)-x+1;
            flag=1;
```

```
        end
    end
    flag=0;
end
sample=list(1,2:point+1);
end
```

5.ifsame.m

```
function flag = ifsame( list,x )
flag=0;
for i=1:1:6
    if list(1,i)==x
        flag=1;
    end
end
end
```

6.mycurverfitting

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

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

end
```

7.randpop

```
function sample = randpop(bounds,popsize,num)
size=0;
while size<popsize
    list=randperm(bounds);
    sample(size+1,:)=list(1,1:num);
    size=size+1;
end
end
```

8.repopulate.m

```
function population = repopulate(population,popsize,point)
for q=1:1:5
    mut(q,:)=population(power(2,q-1),:);
end
k=1;
for i=1:1:50
    population(i,:)=gerand(mut(k,:),point);
    k=ceil(i/10);
```

```
end
for p=51:1:popsize
    population(p,:)=randpop(51,1,point);
end
for j=1:10:41
    population(j,:)=mut(ceil(j/10),:);
end
end
```

9.selection.m

```
function population = selection(fitvalue,population,popsize,point)
population=sort(population,2);
population(:,point+1)=fitvalue;
population=sortrows(population,point+1);
disp(population(1:5,point+1));
population=population(1:popsize,1:point);
disp(population(1:5,:));
end
```

10.test_ur_answer.m

```
%%%%%%% 答案检验程序 2014-11-24 %%%%%%%%

my_answer=[ 3,11,21,30,40,49 ];%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('20141010dataform.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
    % 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_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;

% 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);
```