# 统计推断在数模转换系统中的应用

第 22 组 于逸尘 5140309200 杨嘉诚 5140309198

摘要：本文结合统计方法，由已知数据为基础，对监测某项与外部环境有关的物理量的模块设计了一种成本合理的传感特性校准（定标工序）方案。文中先通过对提供数据采用不同的拟合方式，选取其中较优的一种。然后选择拟合算法，以保证结果的准确性。最终，我们选择了遗传算法
关键词：统计推断、占空比、多项式拟合，遗传算法，Matlab

**Application of Statistical Inference in AD&DA Inverting System**

group number:22

ABSTRACT: In this paper, combined with statistical method, by the basis of the known data, to monitor a physical quantity related to the external environment of the sensing properties of module design a reasonable cost calibration (calibration process). In this paper, we first by fitting to provide data in different ways, the selection of a better one. Then select fitting method, in order to assure the accuracy of the results. In the end, we choose the genetic algorithm
Key words: statistic Interference、polynomial fitting、Genetic Algorithm、Matlab

## 1.引言

在工程实践中往往会设计许多的测量工具，现在我们给出一个例子：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。
为求定标精确，我们往往希望能通过大量数据的拟合来找出最优方案，但这显然在时间和成本上是无法实现的，因此人们希望能够找到一个省时省力的方法来对这些仪器进行定标，在本文中对拟合方式，拟合算法进行讨论，并选取一个较为合理的方法来计算输入-输出关系是，显然启发式搜索总能在这种数据量大的计算当中起到十分重要的作用，本文就是用了启发式搜索的一种——遗传算法来对数据进行拟合。

## 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

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

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。

- 校准方案总成本

按式（3）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。

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

总成本较低的校准方案，认定为较优方案。

# 3、探索拟合算法

3.1 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 遗传算法（GA）

3.2.1 算法概述

遗传算法（Genetic Algorithm）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。遗传算法是从代表问题可能潜在的解集的一个种群(population)开始的，而一个种群则由经过基因(gene)

编码的一定数目的个体(individual)组成。每个个体实际上是染色体(chromosome)带有特征的实体。染色体作为遗传物质的主要载体，即多个基因的集合，其内部表现（即基因型）是某种基因组合，它决定了个体的形状的外部表现，如黑头发的特征是由染色体中控制这一特征的某种基因组合决定的。因此，在一开始需要实现从表现型到基因型的映射即编码工作。由于仿照基因编码的工作很复杂，我们往往进行简化，如二进制编码，初代种群产生之后，按照适者生存和优胜劣汰的原理，逐代（generation）演化产生出越来越好的近似解，在每一代，根据问题域中个体的适应度（fitness）大小选择（selection）个体，并借助于自然遗传学的遗传算子（genetic operators）进行组合交叉（crossover）和变异（mutation），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（decoding），可以作为问题近似最优解。

3.2.2 基本运算过程如下：

a)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)。

b)个体评价：计算群体 P(t)中各个个体的适应度。

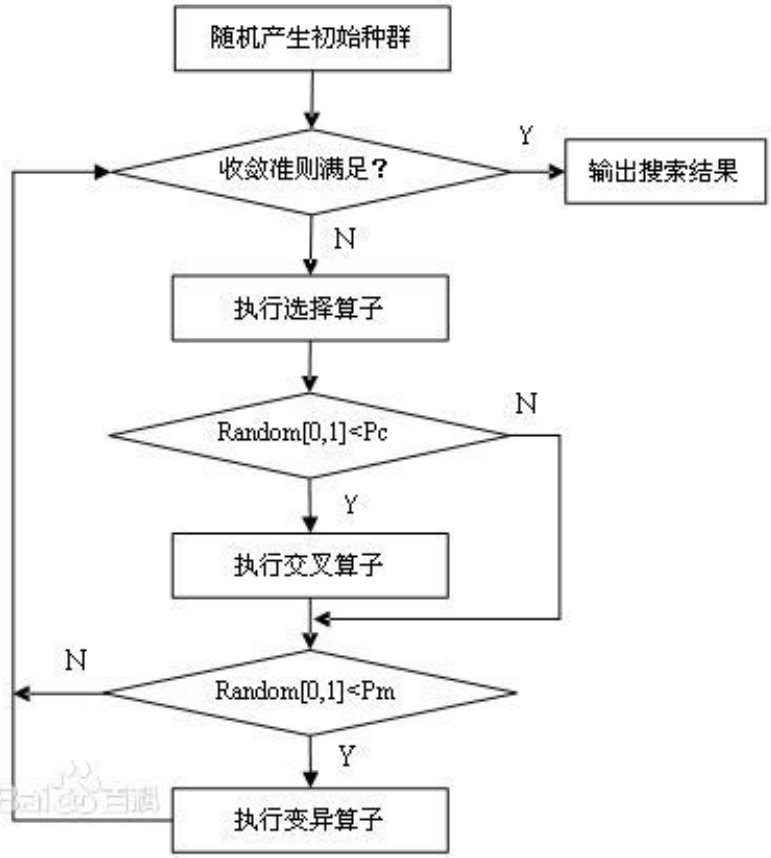
c)选择运算:将选择算子作用于群体。选择的目的是把优化的个体直接遗传到下一代或通过配对交叉产生新的个体再遗传到下一代。选择操作是建立在群体中个体的适应度评估基础上的。

d)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。

e)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。

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)。

f)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。




3.2.3 遗传算法在本案例中的实现

a)初始化：生成 51 位全为 0 的向量，之后将其中随机的位变为 1。考虑到数据的规律性较强，我们初始点的选取个数为随机 4 至 10。

b)个体评价：该操作即从前代种群中选择个体到下一代种群的过程。一般是用个体适应度的分布来选择个体，每个个体按照其适应度占据一个轮盘的部分面积。随机转动一下转盘，当转盘停止转动时，指针所指向的个体即为被选中。由于这个过程中可能将适应度最高的个体排除，因此我们采用精英优势，即每一代适应度最高的个体以及第二高的个体将强制保留至下一代。

c)交叉运算：交叉操作是对任意两个个体进行的。随机选择两个个体，然后按交叉概率对这两个个体进行交叉。如果需要进行交叉，再随机选择交叉位置，将交叉位置以后的二进制串进行对换。（我们设交叉概率为 1）

d)变异运算：根据生物学，遗传时不但有交换染色体，还有变异的过程。取一个较小的变异概率，若一个个体需要进行变异，则在一个随机位置对其及部分相邻的二进制数字进行改变。为了加快求解过程，我们将对较为弱势的个体进行较多的变异。（我们设变异概率为 0.9）

3.2.4 代码实现

见附录

## 3、探索拟合方式

由传感特性图示可得，整个 x-y 曲线成线性关系，因此我们不妨使用多项式拟合来探索拟合方式。我们采用了三次样条插值法和 Hermit 插值法进行尝试，找出最适应的拟合方法和取点方案。

### 3.1 三次样条插值

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。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 等都把非扭结边界条件作为默认的边界条件。

用三次样条插值进行取点（4 个）并计算成本结果如下表。

| 次数 | 1 | 2 | 3 |
|---|---|---|---|
| 采点 | 8,17,29,48 | 3,20,36,49 | 8,17,29,49 |
| 成本 | 140.54 | 105.28 | 140.54 |

### 3.2 Hermit 插值

许多实际插值问题中，为使插值函数能更好地和原来的函数重合，不但要求二者在节点上函数值相等，而且还要求相切，对应的导数值也相等，甚至要求高阶导数也相等。这类插值称作切触插值，或埃尔米特(Hermite)插值。满足这种要求的插值多项式就是埃尔米特插值多项式。

用 Hermite 插值进行取点（4 个）并计算成本结果如下表。

| 次数 | 1 | 2 | 3 |
|---|---|---|---|
| 采点 | 10,29,41,50 | 12,26,43,47 | 2,3,11,22 |
| 成本 | 212.76 | 215.91 | 479.73 |

### 3.3 两种插值方法的对比

三次样条插值把曲线分为几段进行插值，次数为三次；而 Hermit 插值并不进行分段。从样本图像上来看，曲线很明显地分为三段，因此三次样条插值成本显著低于 Hermit 插值成本。

## 4、最终结论

利用遗传算法，并使用三次样条插值法进行拟合后，找到 4 个点分别为 3、20、36、49，使成本最低，最低成本为 105.28；如使用 Hermit 插值法进行拟合，找到 4 个点分别为 10、29、41、50，使成本最低，最低成本为 212.76，与三次样条插值法所得结果差距较大。

综上所述，所选取的 4 个点为 3、20、36、49，最低成本为 105.28。

成本值依然较高，不够理想，初步推断是由于取点较少所致。

## 5、致谢

感谢袁焱老师和李安琪老师的细致讲解、耐心指导，在第一次面谈时提出了宝贵的意见，为我们指正了工作方向。

## 6、参考文献

[1] 武爱文,冯卫国，卫淑芝，熊德文等. 概率论与数理统计[M]. 上海:上海交通大学出版社,2011:250-251.
[2]上海交大电子工程系：统计推断讲座 1～3ftp://202.120.39.248.
[3]百度百科 词条 遗传算法
[4]互动百科 词条 三次样条插值、Hermit 插值

## 6、附录

### 附录 1： 全局函数

```
global x;
global y;
data=csvread('C:\Users\Administrator\Desktop\20150915dataform.csv');
y=zeros(400,51);
x=data(1,:);
for i=1;400
 y(i,:)=data(2*i,:);
end

[m,n,p]=GeneticAlgorithm(50,51,50,1,0.9);
disp ”最优个体”
m
disp ”最优适应度”
n
disp ”得到最优结果的代数”
P
```

### 附录 2： 遗传算法主函数

```
function[m,n,p]=GeneticAlgorithm(pop_size,chromo_size,generation_size,cross_rate,mutate_rate)
```

```
global G;%遗传代数
global fit_value;%当前代适应度矩阵
global best_fit;%最佳适应度
global fit_plot;%各代最佳适应值
global best_individual;%最佳个体
global best_generation;%最佳代数

G=0;
fit_plot=zeros(generation_size,1);

fit_value(pop_size)=0.;
best_fit=99999.9;
best_generation=0;
init(pop_size,chromo_size);%初始化
for G=1:generation_size
    fit(pop_size,chromo_size);%计算适应度
    rank(pop_size,chromo_size);%按适应度进行排序
    choose(pop_size,chromo_size);%选择操作
    cross(pop_size,chromo_size,cross_rate);%交叉操作
    mutate(pop_size,chromo_size,mutate_rate);%变异操作

    %显示每一代过程中的最优适应度
    G
    best_fit
end
plotG(generation_size);%打印算法迭代过程
m=best_individual;%获得最佳个体
n=best_fit;%获得最佳适应度
p=best_generation;%获得最佳个体出现代

clear i;
clear j;
```

## 附录 3: 初始化函数

```
%pop_size:种群大小
%chromo_size:染色体长度

functioninit(pop_size,chromo_size)
global pop;

pop=zeros(pop_size,chromo_size);
for i=1:pop_size
    r=randperm(chromo_size);
```

```
    for j=1:round(6*rand+7)
    %由于数据规律性，初始点选取 4 至 10 个
    pop(i,r(j))=1;%随机进行初始化二进制数
    end
end

clear  i;
clear  j;
clear  r;
```

附录 4: 适应度函数

```
%pop_size:种群大小
%chromo_size:染色体长度

functionfit(pop_size,chromo_size)
global fit_value;
global pop;
```

附录 5:成本计算函数

```
for i=1:pop_size
    fit_value(i)=mycost(pop(i,:),chromo_size);
end

clear  i;
clear  j;
end

%根据输入的二进制向量解码并计算成本

function [p]=mycost(t,chromo_size)
global x;
global y;

p=0;
m=find(t);
n=length(m);

%若取的数据点不多于 3 个则判定为无效
if n<=3
   p=9999;
   return;
end

u=zeros(1,n);
```

```
v=zeros(400,n);
vv=zeros(400,chromo_size);

%解码过程
for i=1:n
    u(1,i)=4.9+0.1*m(1,i);%对 X 的解码
    for j=1:400
    v(j,i)=y(j,m(1,i));%对 Y 的解码
    end
end

%拟合并判断各点成本
for i=1:400
    vv(i,:)=pchip(u,v(i,:),x);%Hermite 插值
    %vv(i,:)=spline(u,v(i,:),x);三次样条插值
    for j=1:chromo_size
    p=p+dotpay(y(i,j),vv(i,j));
    end
end
p=p/400+12*n;

clear i;
clear j;
clear n;
clear m;
```

## 附录 6: 判断两点成本的函数

```
function[pay]=dotpay(a,b)
if abs(a-b)<=0.4
   pay=0;
else
    if abs(a-b)<=0.6
        pay=0.1;
    else
        if abs(a-b)<=0.8
            pay=0.7;
        else
            if abs(a-b)<=1
                pay=0.9;
            else
                if abs(a-b)<=2
                    pay=1.5;
                else
```

```
                    if abs(a-b)<=3
                        pay=6;
                    else
                        if abs(a-b)<=5
                            pay=12;
                        else
                            pay=25;
                        end
                    end
                end
            end
        end
    end
end
```

附录 7: 选择函数

```
%pop_size:种群大小
%chromo_size:染色体长度

functionchoose(pop_size,chromo_size)
global pop;
global fit_value;

pop_new=zeros(pop_size-1,chromo_size);
%轮盘操作
chance=zeros(1,pop_size);
disk=zeros(1,pop_size);
uu=fit_value(pop_size);

%将成本最高的值减去个体成本作为每个个体的适应度
for i=1:pop_size
    chance(i)=uu-fit_value(i);
end
disk(1)=chance(1);
for i=2:pop_size
    disk(i)=disk(i-1)+chance(i);
end

%保留成本最低的两个个体
for i=1:pop_size-2
    idx=1;
    r=rand*disk(pop_size);
    while disk(idx)<r
    idx=idx+1;
```

```
    end;
    for j=1:chromo_size
    pop_new(i,j)=pop(idx,j);
end
end

%将选中的个体代替上一代
for i=3:pop_size
    for j=1:chromo_size
    pop(i,j)=pop_new(i-2,j);
    end
end

clear i;
clear j;
clear k;
clear pop_new;
clear idx;
clear m;

%对个体按适应度大小进行排序，并且保存最佳个体
%pop_size:种群大小
%chromo_size:染色体长度

functionrank(pop_size,chromo_size)
global fit_value;%个体适应度
global fit_plot;%适应度作图
global best_fit;%最佳适应度
global best_individual;%最佳个体
global best_generation;%最佳代数
global pop;
global G;%G 指遗传代数

tmp(chromo_size)=0;

%按照从小至大排序
for i=1:pop_size
    min=i;
    for j=i+1:pop_size
    if fit_value(j)<fit_value(min);
    min=j;
    end
end
if min~=i
```

```
    temp=fit_value(i);
    fit_value(i)=fit_value(min);
    fit_value(min)=temp;
    for k=1:chromo_size
        tmp(k)=pop(i,k);
        pop(i,k)=pop(min,k);
        pop(min,k)=tmp(k);
    end
end

end

%计算最低适应度，用于作图
fit_plot(G)=fit_value(1);

%选取精英个体
if fit_value(1)<best_fit
    best_fit=fit_value(1);
    best_generation=G;
    for j=1:chromo_size
    best_individual(j)=pop(1,j);
    end
end


clear i;
clear j;
clear k;
clear min;
clear temp;
clear tmp;
```

附录 8: 交叉函数

```
%pop_size:种群大小
%chromo_size:染色体长度
%cross_rate:交叉概率

functioncross(pop_size,chromo_size,cross_rate)
global pop;

i=2;
while i<pop_size
    if (rand<cross_rate)
        %防止优秀个体与过于差的个体交叉
```

```
        u=round(rand*2*(pop_size-i)/3);

        if u==0
            continue;
        end
        cp=round(rand*chromo_size);
        if (cp==0||cp==1||cp==pop_size||cp==pop_size-1)
            continue;
        end

        %若交换的基因片段相同则重新进行交叉
        if pop(i,cp:chromo_size)==pop(i+u,cp:chromo_size)
            continue;
        end

        %交换基因片段
        for j=cp:chromo_size
            temp=pop(i,j);
            pop(i,j)=pop(i+u,j);
            pop(i+u,j)=temp;
        end
    end
    i=i+1;
end


clear i;
clear j;
clear temp;
clear cp;
```

附录 9: 变异函数

```
%pop_size:种群大小
%chromo_size:染色体长度
%mutate_rate:变异概率
functionmutate(pop_size,chromo_size,mutate_rate)
global pop;

for i=2:pop_size
if rand<mutate_rate

%随机选择一个基因
mutate_pos=randperm(chromo_size);
```

```
%对于弱势个体，进行较多的变异
for j=1:2+3*(i>pop_size/2)
pop(i,mutate_pos(j))=1-pop(i,mutate_pos(j));
end
end
end
clear i;
clear mutate_pos;
```

## 附录 10:答案检验程序

```
%%%%%%% 答案检验程序 2015-11-04 %%%%%%%%

my_answer=[12,26,43,47 ];%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);
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_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);
```

## 附录 11:答案检验程序 2

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

x=[5.0:0.1:10.0];

% 将你的定标计算方法写成指令代码，以下样式仅供参考
y1=pchip(x_premea,y0_premea,x);%hermit 插值
%y1=spline(x_premea,y0_premea,x);%三次样条插值


end
```