统计推断在数模转换系统中的应用

组号：45　高健宁 5140309270 肖翔志 5140309210

摘要：本文结合统计方法，根据已知的少量数据实现对其余更多的未知数据的推断。文中分 析了不同的拟合方法及其比较，选取了一种较优的拟合方式，并采用遗传算法，模拟一定种 群大小的生物遗传进化方式。通过产生新解，不断比较优化，最终得到比较优化的解。

关键词：模拟退火算法、遗传算法、三次样条插值法、多项式拟合法

Application of Statistical Inference in AD&DA Inverting System

group number:45

ABSTRACT: This report sets up a mathematical model is combined with statistical methods to infer the large amount of unknown data based on a small amount of known data. This report analysis different fitting methods to select an optimum fit and uses Genetic Algorithm to simulate the process of solid cooling. By constantly generating new solutions and optimizing them, we ultimately get more optimal solution.

KEYWORD：Simulated Annealing、Genetic Algorithm、Cubic Sp line Interpolation、Polynomial fitting

1.引言

在工程实践中，对某些物理量（如温度、压力、光强等）的测量需要用到特殊的测量工 具，这些测量工具主要由传感器组成。本次实验的课题为：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。为了解决这一问题我们首先初步分析了实验数据，然后我们选择了遗传算法作为启发式搜索的方法，测算成本并记录，确定 7 个数据点。最后选择三项插值法作为拟合的方法，最后将成本最低的方案最为最后的结果。

2. 评价标准的构建

为评估和比较不同的校准方案，特制定以下成本计算规则。

2.1 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。

2.2 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。

2.3 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。

2.4 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。

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

总成本较低的校准方案，认定为较优方案。

3. 拟合方法的确定

3.1 多项式式拟合分析

(1)一段多次多项式拟合

多项式拟合是通过多项式来拟合曲线，使得残差平方和最小的方式，伪代码为：

①产生一组数据中 n 个随机特征点；

②通过 matlab 多项式拟合函数 p=polyfit（x,y,k）得到 k 次拟合曲线的系数 si；

③通过函数 y2=polyval（p,x1）获取全部数据的拟合值，再求评价分数（去掉每组 的第一个和最后一个数据）；

④令 m=3、4、5，重复上述过程[2]。

(2) 三段多次多项式拟合

伪代码：

①获取一组数据（需判断此组数据未经使用）的 9 个特征点，且按照 3 个不同的分段分别放在三个数组中；

②通过 matlab 多项式拟合函数 p=polyfit（x,y,k）得到三段 k 次拟合曲线的系数 si；

③通过函数 yy1=polyval（p,x1）获取全部数据的拟合值，再求评价分数；

④令 m=3、4、5，重复上述过程[3]。

3.2 插值拟合分析

由于多项式拟合拟合次数达到五次前成本随着次数的增大而减小，但是大于五次时拟 合出的曲线发生严重扭曲，所以只能选择五次拟合但明显成本还是不够低，所以选择了插 值拟合分析。

观察到数据的中间部分线性关系比较明显，两端线性关系比较弱，所以可以用三次样 条插值法来进行拟合。

用 matlab 实现插值拟合，这次实验中我们使用 interp1(x0,y0,xi,‘spline’) 来实现拟合，伪代码为：

（1）产生一组数据中 n 个随机特征点；

（2）通过 matlab 函数 yi=(x,y,x1,‘spline’)获取全部数据的拟合值，再求评价分数（去 掉每组的第一个和最后一个数据）；

（3）处理所有数据，最后得出平均评价分数。 通过比较可以得出三次样条插值拟合方式更好，虽然耗时较长，计算更繁琐，但是其 平均得分和样本都优于多项式拟合。

通过比较可以得出三次样条插值拟合方式更好，虽然耗时较长，计算更繁琐，但是其 平均得分和样本都优于多项式拟合。本实验采用这种拟合方法

4.启发式搜索的选择

4.1 暴力穷举

这个方法血药尝试约 1.54 亿钟不同的组合，显然不切实际，不能采用。

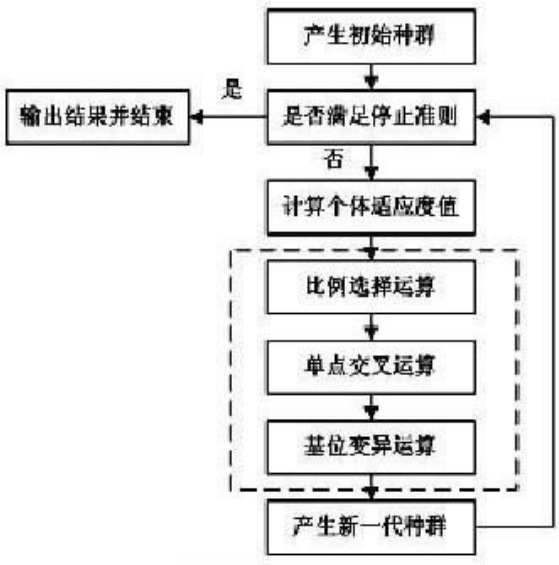
4.2 遗传算法

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。遗传算法是从代表问题可能潜在的 解集的一个种群开始的，而一个种群则由经过基因编码的一定数目的个体组成。每个个体 实际上是染色体带有特征的实体。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。初代种群产生之后，按照适者生存和优胜劣汰的原理，逐 代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大小选择个体， 并借助于自然遗传学的遗传算子进行组合交叉和变异，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。 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。

5.遗传算法的实现

(1)建初始状态 初始种群是从解中随机选择出来的，将这些解比喻为染色体或基因，该种群被称为第 一代，这和符号人工智能系统的情况不一样，在那里问题的初始状态已经给定了。
(2)评估适应度 对每一个解(染色体)指定一个适应度的值，根据问题求解的实际接近程度来指定(以便 逼近求解问题的答案)。不要把这些“解”与问题的“答案”混为一谈，可以把它理解成为要得到答案，系统可能需要利用的那些特性。
(3)繁殖 繁殖(包括子代突变)带有较高适应度值的那些染色体更可能产生后代(后代产生后也将 发生突变)。后代是父母的产物，他们由来自父母的基因结合而成，这个过程被称为“杂交”。
(4)下一代 如果新的一代包含一个解，能产生一个充分接近或等于期望答案的输出，那么问题就 已经解决了。如果情况并非如此，新的一代将重复他们父母所进行的繁衍过程，一代一代 演化下去，直到达到期望的解为止。



遗传算法流程图

6.最终结论

综上所述，利用遗传算法，并使用三次样条插值拟合进行拟合后，可以找到六个点分别 为 3,12,22,31,41,50 使得成本最低，为 92.96，满足评价条件。同时，减少一个点，也可 以找到五个点分别为 3,14,26,39,49 使得成本为 103.56，明显高于前一个选点的成本。而四个点则是成本过大，因此，六个点为宜。从中选择成本最低的点可以在工程上减少检测成本。

选择的点: 3,12,22,31,41,50

7.致谢

感谢袁炎老师和李老师的细心指导和讲解，感谢实验中其他小组的帮助与讨论。

8. 参考文献

8.1 上海交通大学统计推断课程 课件 ppt
8.2 百度词条
8.3 上海交通大学 2013 级第十组实验报告
8.4 百度百科 matlab 基本编程

9. 附录

9.1 遗传算法主函数

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

global G ;
global    fitness_value;
global     best_fitness;
global      fitness_avg;
global  best_individual;
global best_generation;

fitness_avg = zeros(generation_size,1);

fitness_value(pop_size) = 0;
best_fitness      =      0;
best_generation = 0;
initialize(pop_size, chromo_size);

for              G=1:generation_size
    fitness(pop_size,   chromo_size);
    rank(pop_size, chromo_size);
    selection(pop_size,    chromo_size,    elitism);
    crossover(pop_size,  chromo_size,  cross_rate);
    mutation(pop_size, chromo_size, mutate_rate);
end

plotGA(generation_size);
m = best_individual;
n = best_fitness;
p = best_generation;

q = [];
for j=1:chromo_size
    if best_individual(j) == 1
          q = [q, j];
```

```
    end
end

m
n
p
q

clear i;
clear j;
```

9.2 遗传算法主要函数实现

(1)计算种群适应度

```
function fitness(pop_size, chromo_size)
global fitness_value;
global pop;
global G;

for          i=1:pop_size
    fitness_value(i) = 0;
end

data=csvread('20141010dataform.csv');
a=1:2:(length(data)-1);
b=2:2:length(data);
X=data(a,:);
Y=data(b,:);

for  i=1:pop_size
    point = [];
    for j=1:chromo_size if
        pop(i, j) == 1
            fitness_value(i) = fitness_value(i) + 12 * length(a);
            point = [point, j];
        end
    end

    if length(point) < 3
        fitness_value(i) = fitness_value(i) + 10000;
    end

    point_x = X(:,point);
    point_y = Y(:,point);
```

```
    for q=1:length(a)
        point_x_q  =  point_x(q,:);
        point_y_q = point_y(q,:);



        point_x_q_all  =  X(q,:);
        point_y_q_all = Y(q,:);

        y_current = spline(point_x_q, point_y_q,point_x_q_all);

        Q = abs(point_y_q_all - y_current);

        for j=1:51
            if Q(j) > 5
                fitness_value(i) = fitness_value(i) + 25;
                else if Q(j) > 3
                        fitness_value(i) = fitness_value(i) + 12;
                        else if Q(j) > 2
                                fitness_value(i) = fitness_value(i) + 6;
                                else if Q(j) > 1
                                        fitness_value(i) = fitness_value(i)
+ 1.5;
                                    else if Q(j) > 0.5
                                            fitness_value(i) =
fitness_value(i) + 0.5;
                                        else if Q(j) >= 0
                                                fitness_value(i) =
fitness_value(i) + 0;
                                            end
                                    end
                                end
                        end
                end
            end
        end
    end


    fitness_value(i)  =  fitness_value(i)  /  469;
    fitness_value(i) = 100000 - fitness_value(i);
end


clear i;
clear j;
```

(2)选择操作

```
function selection(pop_size, chromo_size, elitism)
global pop;
global fitness_table;

for i=1:pop_size
    r = rand * fitness_table(pop_size);
    first = 1;
    last = pop_size;
    mid = round((last+first)/2);
    idx = -1;
    while (first <= last) && (idx == -1)
        if r > fitness_table(mid)
            first = mid;
        elseif r < fitness_table(mid)
            last = mid;
        else
            idx = mid;
            break;
        end
        mid = round((last+first)/2); if
        (last - first) == 1
            idx = last;
            break;
        end
   end

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

if elitism
    p = pop_size-1;
else
    p = pop_size;
end
for i=1:p
   for j=1:chromo_size
        pop(i,j) = pop_new(i,j);
   en
d end

clear i;
```

```
clear j; clear
pop_new;  clear
first;    clear
last; clear idx;
clear mid;
```

(3)单点变异

```
function mutation(pop_size, chromo_size, mutate_rate)
global pop;

for i=1:pop_size
    if rand < mutate_rate
        mutate_pos = round(rand*chromo_size); if
        mutate_pos == 0
            continue;
        end
        pop(i,mutate_pos) = 1 - pop(i, mutate_pos);
    end
end

clear i;
clear mutate_pos;
```

(4)单点交叉

```
function crossover(pop_size, chromo_size, cross_rate)
global pop;
for i=1:2:pop_size if(rand
    < cross_rate)
        cross_pos = round(rand * chromo_size); if
        or (cross_pos == 0, cross_pos == 1)
            continue;
        end
        for   j=cross_pos:chromo_size
            temp = pop(i,j); pop(i,j)
            = pop(i+1,j); pop(i+1,j) =
            temp;
        end
    end
end



clear i;
clear j;
```

```
clear temp; clear cross_pos;
```

9.3 拟合函数

```
function Result= fitness( ~,Result ,Gene)
global originY;
global x;
global tempMaxChosenY;
tmp=12*51+400*51*25;
m=1;
[Xtemp,Ytemp]=find(Gene(m,:)~=0);
count=length(Xtemp);
tempX=Ytemp.*0.1+4.9;
ChosenY=originY(:,Ytemp);
%calculateY=spline(tempChosenX,ChosenY,x); %采用三次样条插值方法
calculateY=pchip(tempX,ChosenY,x);  %分段三次 hermite 插值法
%成本计算函数
errorCount=abs(calculateY-originY);
le0_4=(errorCount<=0.4);
le0_6=(errorCount<=0.6);
le0_8=(errorCount<=0.8);
le1_0=(errorCount<=1);
le2_0=(errorCount<=2);
le3_0=(errorCount<=3);
le5_0=(errorCount<=5);
g5_0=(errorCount>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);
Result(1,m)=sum(si)/400+12*count; %计算误差成本和测定成本
tempM=tempX;
tempMaxChosenY=Ytemp;
tempMaxChosenY;
end
```