统计推断在数模转换中的应用

组号：52　徐英展（组长）：5140309420　杨文泰：5140309425

摘要：

统计推断根据带随机性的观测数据（样本）以及问题的条件和假定（模型），而对未知事物做出的，以概率形式表述的推断。在本课程中，我们使用 Matlab 寻求实现通过几个点拟合出一条曲线的优化方法，并通过遗传算法进行启发式搜索，解决一个组合优化问题的思路和具体操作。具体所需的样本有老师提供，一共 400 组数据，每组数据各有 51 个数据点。

关键词：

统计推断，Matlab，数模转换，启发式搜索，遗传算法

Summary：

Statistical inference based on the observation data (sample) and the conditions and assumptions (models) of the problem. In this course, we use Matlab to find a way to achieve the optimization of a curve through a few points, and through the genetic algorithm to search for a combination of optimization problems and the idea of specific operation. The sample provided by the sample is provided by the teacher, a total of 400 sets of data, each of which has 51 data points.

Key word:

Statistical inference, Matlab, data transformation, heuristic search, genetic algorithm

1. 引言

1.1 课题概述

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。[1] 实验共 400 组数据，每组数据各有 51 个数据点。

1.2 课程基本问题

由于实验数据较多，因此分析处理工作量很大。所以选择某一些需要进行查找的个体，通过有线次数的你和判定，估计对象值与数据之间的特性关系。然后再进行模拟运行，以求出最小总成本。

2. 数据统计与初步分析

2.1 问题的引入

我们要为某型产品内部的一个监测模块，寻求校准工序的优化方案，为 X 和 Y 间的关系进行校准定标。我们把 X 为某个特定值时，针对相应 Y 的数值进行的测量过程，称作一次测定。测定要付出一定的成本；定标误差也要折算成一定的成本。

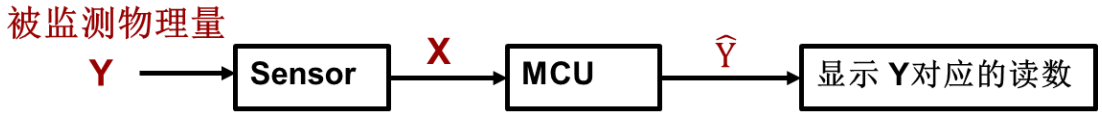

图 1 检测模块工作流程[2]

对于定标总成本：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。总成本最低的方案，对特定样品而言不一定是最低成本。

2.2 初步数据分析

2.2.1 数据初步拟合实验

对于 400 组数据中的一个典型样本进行观察分析之后可以发现：

（1）特性曲线都是单调递增的。

（2）特性曲线大致可以分为首，中，尾三部分，这三部分虽然都不是线性关系的，但是都有其独特的曲率变化模式。所以可以对于这三段分别进行取样，之后综合起来判断整体趋势。

（3）个体产品的特性曲线形态相似但是两两不同。

2.2.2 选择不同的拟合方法

（1）多项式拟合法：

设一条三次多项式曲线为：

```
y=a3*(x^3)+a2*(x^2)+a1*x+a0
```

随机选取七个点进行多项式拟合，通过某一样本个体的定标成本函数：

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

其中单点定标误差成本给出（见附录），结合校准方案总体成本函数

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

拟合计算出平均成本。

我们进行了三组的数据，分别运行三代进行实验验算之后:

表 1 多项式拟合平均成本

[1 9 23 27 40 48]	cost=122.791250
[3 14 22 28 38 49]	cost=109.630000
[2 7 17 28 40 48]	cost=109.630000

（2）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。样条曲线是由分段三次曲线并接而成，在连接点上二阶导数连接。

使用 test 文件运行同样的三组数据之后发现：

表 2 三次样条插值法平均成本

[1 9 23 27 40 48]	cost=107.29
[3 14 22 28 38 49]	cost=107.18
[2 7 17 28 40 48]	cost=102.56

由于最后对于同样的几组数据，三次样条插值法所计算出来的最优解成本较小，所以我们选择使用三次样条插值法。

3. 遗传算法的思想与实现

由于种群样本较大，所以我们选择启发式算法来解决这个问题，并选择其中的遗传算法来解决。

3.1 遗传算法简介

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。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。

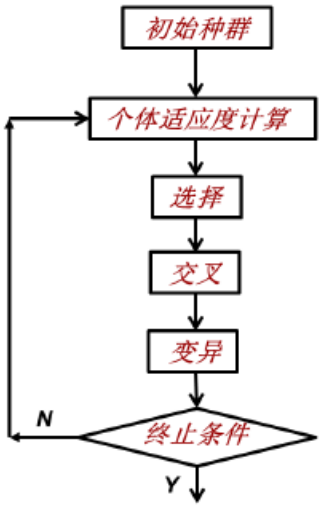

图 2 遗传算法具体流程图

遗传算法的具体流程原理图如上。

3.2 遗传算法的实现

3.2.1 初始群体的生成

初始中区你的规模大小设定为 200。因为 6 的二进制编码转化为十进制之后，最多可以表示 64 个测试点，而我们需要表示的是 51 个测试点，所以我们选择 6 个点。然后将 1~51 这 51 个点的编号转换成 6 的 2 进制编码，在表示出编码对应的 x 值。

3.2.2 适应性评估

根据选定的 6 个点对应的 y 值，用三次样条插值法拟合出 y 的曲线，将拟合值与测定值相比较计算出总的平均成本。对于个体的成本，利用之前所给出的成本计算函数进行求和（利用“成本计算函数”），由于今天成本越低，方案越优，所以我们定义成本的倒数的三次方，作为适应度。

3.2.3 选择过程

针对每一个个体计算起适应度，为达到使其中适应度较高的个体出现的比例较大。对于所有个体的适应度，进行归一化（即对其适应度等比例缩小，使和为一），归一化之后的结果为下一代中出现的比例。如果归一化结果使得下一代出现的数量为非整数，则进行取整处理。超出总体部分由出现概率较小的个体来承担，除去，保证每一代的个体总量均为 200（利用“归一化函数”）。

3.2.4 交叉

3.2.4.1 排序

对于确定了适应度的下一代个体进行有适应度从大到小的排序，且将序号记录在每一个个体的结构数据中。

3.2.4.2 确定交叉节点

由于我们定义的群体总量为 200，为偶数，所以恰好可以是所有都进行交叉。产生 1~200 之间的 1000 个相异随机数，其序号代表的个体被标记选中。

3.2.4.3 进行交叉

对于群体中的每一个个体将其位于交叉结点之后的数据与其交叉对象进行互换（利用“交叉数据函数”）。交叉完成之后的群体作为下一代群体。在交叉运算过程中可能会产生不再 1~51 中的个体，先不做处理，在下一步运行变异运算时淘汰。

3.2.4.4 再次排序

过程同 3.2.4.1.

3.2.5 变异

3.2.5.1 确定是否变异

随机产生 1~300 之间的一个数，若这个数为 300，则进行变异。则变异概率为 1/300.

3.2.5.2 进行变异

一共有 6 个点，每个点均可能产生变异。因为在进行几代的遗传之后，已经收敛的结果对于之后每一代的数据点只是有较小的概率产生较小的波动，所以更加容易产生最优解；如果变异完全随机的话，很容易偏移最优解，因此我们选择变异点固定为二进制的最后一位或者两位（对应十进制的 1~3），采用这种半固定的方式进行变异。

3.2.6 终结判断条件

产生初始群体时，产生一个数据 MARK=0，每进行一次交叉，MARK+1。直到 MARK 大于 50，即进行了 50 代遗传选择，我们认为已经较为接近最优解，可以终止。

3.3 遗传算法的具体运算结果

我们分别计算了选取 5 个点，6 个点，7 个点的遗传 10 代的运算成本：

表 3 取样点个数分别为 5，6，7 的成本比较

[2 13 25 38 48]	cost=106.076250
[3 10 22 31 42 50]	cost=93.941250
[2 10 21 26 33 42 49]	cost=96.053750

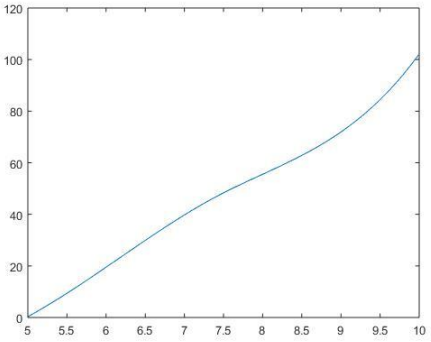

图 3 五个点进行三次样条插值法

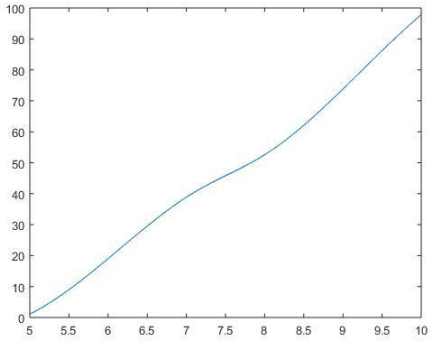

图 4 六个点进行三次样条插值法

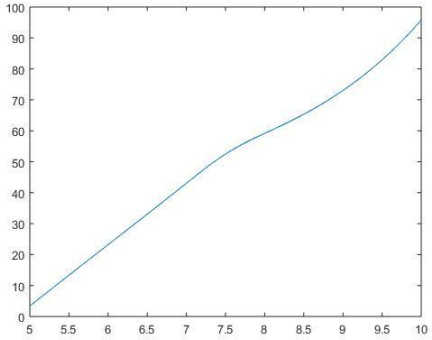

图 5 七个点进行三次样条插值法

经过比较我们可以发现如果选取的点过少，将产生较大的成本偏差；选取 6 个点相比于选取 7 个点成本较小，故而我们选取 6 个点，可以得到较优解。

4. 结论

经过多次尝试，我们组最终得到的选取 6 个点的最优解选点方案是[3 11 22 31 43 50]，最优成本为 93.368750.

5. 参考借鉴说明

本实验初稿其中所用知识思想，借鉴了之前的一些学长报告以及课件中给出的内容，在以下表格中予以声明：

表 4 主要参考声明

主要参考方面	主要参考文献
遗传算法初步思想设计	百度文库“遗传算法及其 MATLAB 实现”
遗传算法具体流程书写格式 代码格式 遗传算法流程说明图	ftp 上的“统计推断-第 20 组（艾玥祺）课程设计报告”书写格式以及主要代码都是在参考之后，结合自己的思想进行模仿
问题引入以及数据初步分析	“统计推断”课程设计的要求 V2.2 2015-9-22”
遗传算法简介	百度百科“遗传算法”
三次样条插值法简介	百度百科“三次样条插值法”

附录;

1. 单点定标误差成本

由以下给出：

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

单点定标误差的成本上式计算，其中 $y_{i,j}$ 表示第 i 个样本之第 j 点 Y 的实测值，$\hat{y}_{i,j}$ 表式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。

2. 代码：

2.1 主运行程序（main.m）

```
data=xlsread('y');

size=50;
point=6;
D=50;
G=1;
xp=fcreatexp(size,point);
while(G<=D)
    code=fdtob(size,point,xp);
    x=fx(size,point,xp);
    y=fy(size,point,data,xp);
    ys=fspl(size,x,y);
    w=fw(size,ys,data);
    A=fA(size,point,w);
    index=findex(size,A);
    fprint(A,index,point,xp,G);
    live=flive(size,point,A,xp);
```

```
        num=fnum(size,live);
        select=fselect(size,point,num,code);

        cross=fcross(size,point,select);

        muta=fmuta(size,point,cross);
        xp=fback(size,point,muta,xp);
        G=G+1;
    end
```

2.2 产生选点方案程序(fcreatexp.m 与 fcreate.m)

```
    function f=fcreatexp(size,point)
    x=zeros(size,point);
    minn=1; maxn=51;
    for i=1:size
        z=fcreate(minn,maxn,point);
        for j=1:point
            x(i,j)=z(j);
        end
    end
    f=x;
    end

    function f=fcreate(minn,maxn,n)
    x=zeros(1,n);
    same=1;
    while same==1
        for i=1:n;
            x(i)=randi(maxn-minn+1)+minn-1;
        end
        same=0;
        for i=1:n-1
            for j=i+1:n
                if x(j)==x(i)
                    same=1;
                end
            end
        end
    end
    for i=1:n-1
        for j=i+1:n
            if x(j)<x(i)
                t=x(j); x(j)=x(i); x(i)=t;
            end
```

```
        end
    end
    f=x;
    end
```

2.3 二进制码转换程序(fdtob.m)

```
    function f=fdtob(size,point,xp)
    c=zeros(size,point,6);
    for i=1:size
        for j=1:point
            t=xp(i,j);
            for k=6:-1:1
                c(i,j,k)=mod(t,2);
                t=floor(t/2);
            end
        end
    end
    f=c;
    end
```

2.4 将选点方案转换为对应的 x 值（fx.m）

```
    function f=fx(size,point,xp)
    x=zeros(size,point);
    for i=1:size
        for j=1:point
            x(i,j)=5+(xp(i,j)-1)*0.1;
        end
    f=x;
    end
```

2.5 取出对应原始数据的 y 值（fy.m）

```
    function f=fy(size,point,data,xp)
    y=zeros(size,400,point);
    for i=1:size
        for j=1:400
            for k=1:point
                y(i,j,k)=data(j,xp(i,k));
            end
        end
    end
    f=y;
    end
```

2.6 计算插值拟合出的 y 标准值(fspl.m)

```
function f=fspl(size,x,y)
ys=zeros(size,400,51);
for i=1:51
    xd(i)=5+(i-1)*0.1;
end
clear i;
for i=1:size
    for j=1:400
        pp=spline(x(i,:),y(i,j,:));
        t=ppval(pp,xd);
        for k=1:51
            ys(i,j,k)=t(k);
        end
    end
end
f=ys;
end
```

2.7 计算总平均误差成本（fw.m）

```
function f=fw(size,ys,data)
w=zeros(1,size);
for S=1:size
    for i=1:400
        for j=1:51
            t=abs(ys(S,i,j)-data(i,j));
            if t<=0.5
                continue;
            end
            if t<=1
                w(S)=w(S)+0.5;
                continue;
            end
            if t<=2
                w(S)=w(S)+1.5;
                continue;
            end
            if t<=3
                w(S)=w(S)+6;
                continue;
            end
            if t<=5
                w(S)=w(S)+12;
                continue;
            end
```

```
                w(S)=w(S)+25;
            end
        end
    end
    for i=1:size
        t(i)=w(i)/400;
    end
    f=t;
    end
```

2.8 计算总平均成本（fA.m）

```
    function f=fA(size,point,w)
    A=zeros(1,size);
    cost=12*point;
    for i=1:size
        A(i)=w(i)+cost;
    end
    f=A;
    end
```

2.9 计算最小值的下标（findex.m）

```
    function f=findex(size,A)
    index=1;
    for i=2:size
        if A(i)<A(index)
            index=i;
        end
    end
    f=index;
    end
```

2.10 打印当前最优解(fprint.m)

```
    function fprint(A,index,point,xp,G)
    fprintf('G=%d cost:%f\n',G,A(index));
    for j=1:point
        fprintf('%d ',xp(index,j));
    end
    fprintf('\n\n');
    end
```

2.11 计算存活概率（flive.m）

```
    function f=flive(size,point,A,xp)
    for i=1:size
        temp(i)=10000/(A(i)^3);
```

```
end
Sum=0;
for i=1:size
    for j=1:i-1
        if temp(i)==temp(j)
            flag=1;
            for k=1:point
                if xp(i,k)~=xp(j,k)
                    flag=0;
                    break;
                end
            end
            if flag==1
                temp(i)=0;
                break;
            end
        end
    end
    Sum=Sum+temp(i);
end
live(1)=temp(1)/Sum;
for i=2:size
    live(i)=temp(i)/Sum+live(i-1);
end;
f=live;
end
```

2.12 计算不同个体被选中的次数(fnum.m)

```
function f=fnum(size,live)
num=zeros(1,size);
for i=1:size
    r(i)=rand;
end
for i=1:size
    j=1;
    if r(i)<=live(1)
        num(1)=num(1)+1;
        continue;
    end
    while r(i)>live(j)
        j=j+1;
    end
    num(j)=num(j)+1;
end
```

```
f=num;
end
```

2.13 计算选择结果（fselect.m）

```
function f=fselect(size,point,num,code)
s=zeros(size,point,6);
i=1;
for j=1:size
    for k=1:num(j)
        for p=1:point
            for q=1:6
                s(i,p,q)=code(j,p,q);
            end
        end
        i=i+1;
    end
end
f=s;
end
```

2.14 交叉（fcross.m）

```
function f=fcross(size,point,select)
cross=zeros(size,point,6);
rd=randperm(size);
for i=1:2:size
    for p=1:point
        r=randi(6);
        for q=1:r
            cross(i,p,q)=select(rd(i),p,q);
            cross(i+1,p,q)=select(rd(i+1),p,q);
        end
        for q=r+1:6
            cross(i,p,q)=select(rd(i+1),p,q);
            cross(i+1,p,q)=select(rd(i),p,q);
        end
    end
end
f=cross;
end
```

2.15 变异（fmuta.m）

```
function f=fmuta(size,point,cross)
for i=1:size
    for j=1:point
```

```
            prob=randi(300);
            if prob==1
                a=4+randi(2);
                if cross(size,j,a)==1
                    cross(size,j,a)=0;
                else
                    cross(size,j,a)=1;
                end
            end
        end
    end
    f=cross;
    end
```

2.16 将二进制转回十进制（fback.m）

```
    function f=fback(size,point,muta,xp)
    t=zeros(size,point);
    for i=1:size
        for p=1:point
            for q=1:6
                t(i,p)=t(i,p)+muta(i,p,q)*(2^(6-q));
            end
        end
    end
    for i=1:size
        for j=1:point
            if t(i,j)>51||t(i,j)<1
                t(i,j)=randi(51);
            end
        end
    end
    for i=1:size
        same=1;
        while(same==1)
            same=0;
            for j=1:point-1
                for k=j+1:point
                    if t(i,j)==t(i,k)
                        same=1;
                        t(i,j)=randi(51);
                    end
                end
            end
        end
```

```
end
for i=1:size
    for j=1:point-1
        for k=j+1:point
            if t(i,k)<t(i,j)
                tmp=t(i,k); t(i,k)=t(i,j); t(i,j)=tmp;
            end
        end
    end
end
for i=1:size
    for j=1:point
        xp(i,j)=t(i,j);
    end
end
f=xp;
end
```

2.17 多项式拟合（测试用）（fploy.m）

```
function f=fpoly(size,x,y)
ys=zeros(size,400,51);
for i=1:51
    xd(i)=5+(i-1)*0.1;
end
clear i;
yt=zeros(1,6);
for i=1:size
    for j=1:400
        for k=1:6
            yt(k)=y(i,j,k);
        end
        pp=polyfit(x(i,:),yt,4);
        t=polyval(pp,xd);
        for k=1:51
            ys(i,j,k)=t(k);
        end
    end
end
f=ys;
end
```