统计推断在数模转换系统中的应用

26 组　李圣杨 5103039060　金沙 5130309606

摘要：在电子产品的监测模块中，为了对被测物理量进行定标，需要了解传感器的特性曲线，研究其输入输出函数关系，唯一的办法就是通过采样进行拟合，然而但采样值和采样总体较大时，这项工艺的耗时会十分庞大。为了解决这一问题，必须通过一个优化采样的启发式算法来解决。本文以遗传算法为基础，讨论了采样方案的优化，运用 MATLAB 等计算工具，初步提出了一种定标方案。

关键词：定标，拟合，启发式算法，遗传算法

1. 引言

传感器元件中普遍存在的输入输出非线性关系要求我们为电子产品提供一种定标工序。由于数据众多，只能以优化采样的启发式算法来解决。本课题中我们尝试多种拟合方案，运用基于遗传算法的搜索算法优化定标方案，尽量减小测量成本与误差成本。通过对课题的研究，我们能够了解目前主流的遗传算法的特性，并且对数据的拟合有更深的理解。

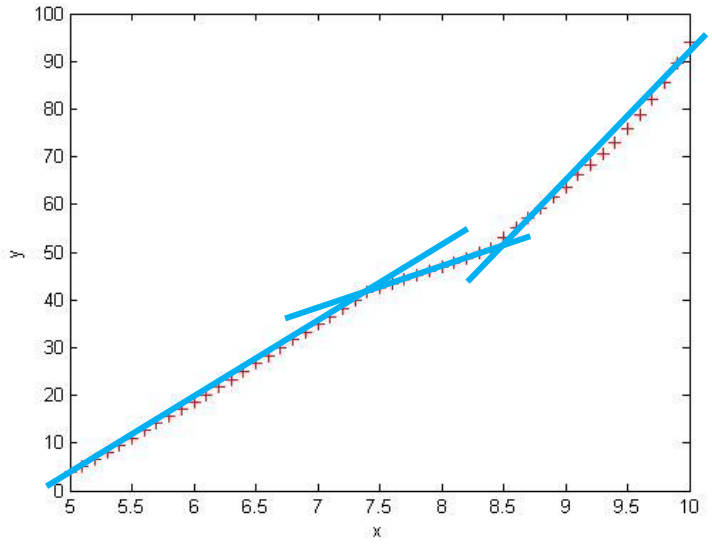


图 1 数据样本的特点

2. 拟合方案的研究

根据数据样本的特点，可对可行的拟合方案进行分析。

2.1 成本计算函数

定标过程中，测定要付出一定成本，定标误差也要换算成一定成本。所以总成本由测量成本和误差成本两部分组成。

单点标定误差成本

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

其中下标 i 代表样本序号，下标 j 代表观测点序号。

对某一样本的定标成本：

$$S_i=\sum_{j=1}^{51}s_{i,j}+20N_i \quad (2)$$

其中第一项是误差成本，第二项是测量成本。Ni 是样本定标时测定的点数。

定标方案总成本：

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

2.2 适应度函数

由于课题目标是找出平均总成本最小的定标方案，而由遗传算法的特征可知，适应度函数应取一与平均总成本函数负相关的函数，因此不妨取为总成本的倒数。即

$$f=\frac{1}{c} \quad (4)$$

2.3 具体的拟合方案

2.3.1 三次多项式拟合

由高等数学知识可知，任意函数均可展开为多项式函数。由教案中所给出的样本特征，图像的斜率由大到小在到大，其导函数是一个下凹的函数，形状与开口向上的抛物线相似，故考虑用三次多项式函数拟合。

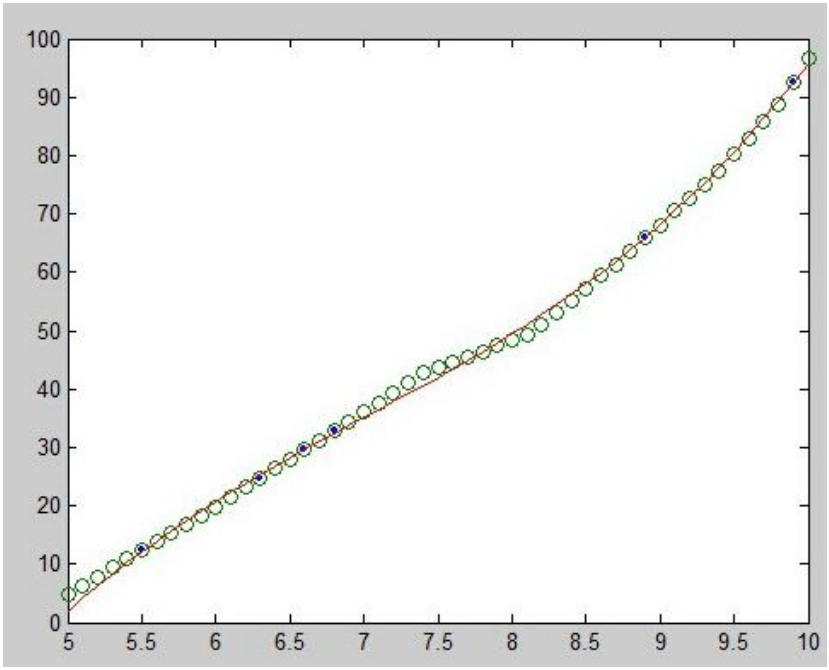


图 2 三次函数拟合结果

由图 2 可见，三次函数基本能够较为准确地描述样本的变化规律，同时运算量并不大，是一种可取的拟合方案。

2.3.2 四次多项式拟合

从以上拟合结果看，三次函数的拟合在局部上存在较大误差，因此我们可以考虑更高次数的多项式拟合，这样虽然函数表达式较为复杂，拟合耗时也较长，但拟合结果更为精确。根据三次函数拟合的结果，我们选择四此函数来进行进一步的实验。

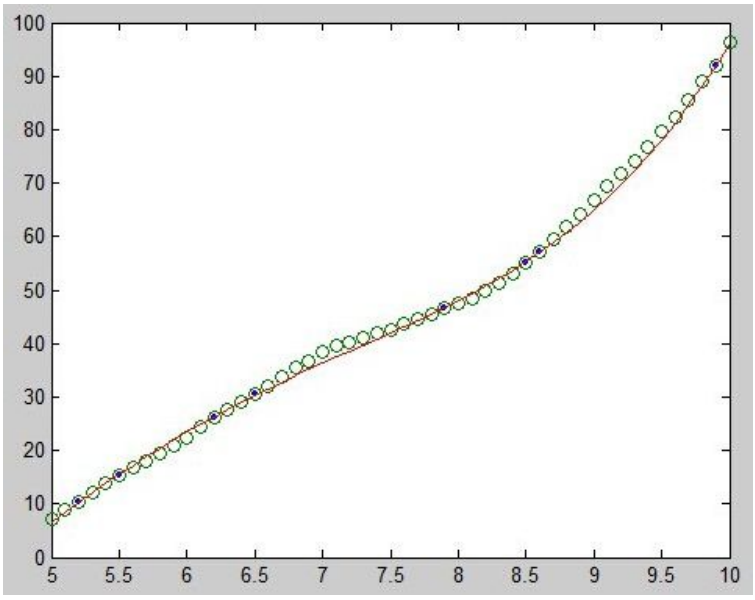


图 3 四次函数拟合结果

与三次多项式拟合相比，四次多项式的拟合更为精确一些，得到的结果也更可靠。但这种拟合的缺点在于运算量较大，运算耗时更长，

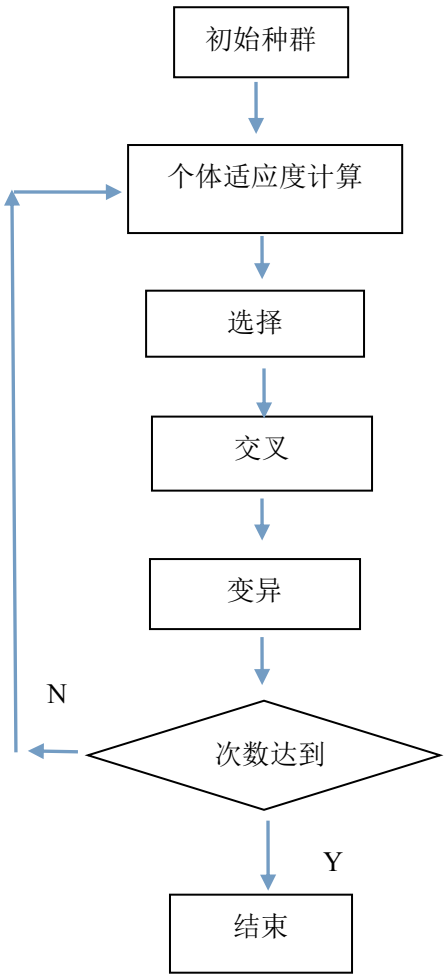
3. 搜索算法的设计

3.1 概述

我们选定使用遗传算法来进行取点方案的搜索。

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。算法中产生个体采用产生 1 至 53 的 7 个随机数的方法，而从父代向子代的遗传主要分三种方式：1，选择（selection），选择父代中优秀的个体直接传递给下一代，即保留父代。2，交叉（crossover），以两个父代的基因为基础，进行交叉重组，生成下一代，好的基因会有更大的概率传给子辈。3，变异（mutation），每个父代个体都有一定的概率发生随机的基因突变。

3.2 流程图



3.3 编码方法

每一个样本共含有 51 个观测点，故取点方案可以用 51 位 01 编码来表示。对一个 51 位二进制数，从左往右第 k 位为 0 表示取点方案中不取第 k 点；若为 1 则取第 k 点。这样就可以把取点方案编码为 51 位二进制数。

3.4 初始种群的产生

遗传算法中的初始种群由随机方式产生。即以 51 个 0,1 随机数组成一个编码来表示一种取点方案。这样重复 30 次，得到 30 组编码，以此作为遗传的初始种群。对问题的解做一些估算，易知拥有最小平均成本的取点方案所取点数不会超过 10，因此在随机构造初始种群时可调整其概率，使初始种群中个体取点方案编码的“1”的数目平均值在 10 左右，这样可以减少迭代次数，提高程序效率。

3.5 交叉算子

根据适应度的高低确定每一个取点方案被选中的概率，然后按概率选出两个取点方案作为父母，随机选择出需交换的基因片段，然后彼此交换该部分基因，得到两个新的取点方案作为后代。例如：

第 m 个取点方案 b[m][51]=[0,0,0,1,1,0,1,……,1,0]

第 n 个取点方案 b[n][51]=[1,1,0,0,0,1,1,……,0,0]

第 q=4 个位置交换

则

b’[m][51]=[0,0,0,1,0,1,1,……,0,0]

b’[n][51]=[1,1,0,0,1,0,1,……,1,0]

3.6 变异算子

在 51 位编码的每一位上都有一定的概率发生变异。

遗传算法引入变异的目的有两个:一是使遗传算法具有局部的随机搜索能力。当遗传算法通过交叉算子已接近最优解邻域时，利用变异算子的这种局部随机搜索能力可以加速向最优解收敛。显然，此种情况下的变异概率应取较小值，否则接近最优解的积木块会因变异而遭到破坏。二是使遗传算法可维持群体多样性，以防止出现未成熟收敛现象。此时收敛概率应取较大值。

3.7 参数设置

popsize =30(每次繁殖交叉的个体数，构成繁殖池。在 16-30 之间，必为偶数)；

pm=0.006；(变异的概率，在 0.005-0.009 之间，较小)；

Loopnum=1000；(遗传代数，过大会增加耗时)；

4. 总结

从以上结果来看，我们设计的拟合方案以及算法是可行的，最后得出的结果总成本较小，基本符合要求。

对于实验中的 938 组数据，我们得到如下结果：、

表 1 两种拟合方案的比较

拟合方式	最优结果（最小成本）	取点集合
三次多项式	103	6,14，17,19,43,50
四次多项式	117	3,6,13,16,30,36,37,50

本课题实验所涉及到的知识主要是利用 matlab 实现大量数据的拟合处理，完成起来比较耗时耗力。这次实验最大的收获就是对遗传算法的深入了解，并掌握了基于遗传算法解决实际问题的方法。

本次课程的实践中，小组合作遇到了很多困难，一开始是方法的选取，后来是遗传算法的生疏，还有 Matlab 软件的运用，在经过不断的学习和探究之后，摸清了遗传算法的脉络，并结合强大的 Matlab 功能完成了多项式拟合法的统计。之后，又一次编写代码优化算法，都得到了不错的结果。

5. 参考文献

[1]段玉倩，贺家李. 遗传算法及其改进. 电力系统及其自动化学报.1998 年 第 1 期

[2]王勇,毕大平. DS 证据理论在雷达体制识别中的应用. 电子对抗技术. 2005 年 11 月 第 6 期

[3]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 ftp://202.120.39.248

6. 附录

本课题中使用的 MATLAB 程序代码如下：

文件 fitness.m:

```
function fit=fitness(u,w,a,cx,cy)
    %% const
    num=30;          %%样本数量
    len=51;          %%样本长度
    n0=469;          %%样本总数
    q0=12;           %%单点测定成本
    ti=4;            %%拟合函数最高次数

    er=0;   %%成本

    for i=1:n0

        %%拟合取点
        b=find(a);
        l=length(b);
        er=er+q0*l;
        p1=polyfit(cx(i,b),cy(i,b),ti);
        ny1=polyval(p1,cx(i,:));

        %%计算误差
        d=abs(cy(i,:)-ny1);
        for j=1:len
            if (d(j)>1 && d(j)<=2)
                er=er+1;
            end
            if (d(j)>2 && d(j)<=3)
                er=er+2;
            end
            if (d(j)>3 && d(j)<=5)
                er=er+4;
            end
            if d(j)>5
                er=er+10;
            end
        end

    end

    %%输出图像
```

```
    if (u==1)
        b=find(a);
        p1=polyfit(cx(w,b),cy(w,b),ti);
        ny1=polyval(p1,cx(w,:));
        plot(cx(w,b),cy(w,b),'.',cx(w,:),cy(w,:),'o',cx(w,:),ny1);
    end

    %%计算适应度
    er=er/n0;
    fit=1/er;

End
```

文件 genetic.m

```
%% const
num=30;          %%样本数量
len=51;          %%样本长度
p0=0.006;        %%变异概率
n0=469;          %%样本总数
k=0;             %%最大适应度样本下标

a=zeros(num,len);
b=zeros(num,len);     %%取点方法
fit=zeros(1,len);     %%适应度数组
pro=zeros(1,len);     %%概率数组
used=zeros(num);      %%判断是否已用过
cx=zeros(num,len);    %%x 轴坐标
cy=zeros(num,len);    %%y 轴坐标

%%打开文件
load('data.in');
fid1=['answer','.out'];
c=fopen(fid1,'w');

%%读入 x 轴坐标和 y 轴坐标
for k=1:n0
    cx(k,:)=data(2*k-1,:);
    cy(k,:)=data(2*k,:);
end

%%利用子函数计算适应度
```

```
for i=1:num
    for j=1:len
        x=rand(1);
        if x<0.2
            a(i,j)=1;
        end
    end
end

for z=1:100

    %%计算适应度
    for i=1:num
        fit(i)=fitness(0,k,a(i,:),cx,cy);
    end

    if z==100
        break
    end

    %%计算概率
    s=0;
    s=sum(fit);

    pro(1)=fit(1)/s;
    for i=2:num
        pro(i)=pro(i-1)+fit(i)/s;
    end

    %%按概率取点并生成新样本
    for i=1:num
        x=rand(1);
        k=1;

        while (x>pro(k))
            k=k+1;
        end
        b(i,:)=a(k,:);
    end

    %%初始化
    used=zeros(1000,1);

    %%交换生成新样本
```

```
    for i=1:num/2
        m=1;
        n=1;
        q=0;

        while (used(m)~=0)
            m=mod(round(rand()*num),num)+1;
        end
        used(m)=1;
        while (used(n)~=0)
            n=mod(round(rand()*num),num)+1;
        end
        used(n)=1;
        q=mod(round(rand()*len),len)+1;

        for j=1:q
            a(2*i-1,j)=b(m,j);
            a(2*i,j)=b(n,j);
        end

        for j=q+1:len
            a(2*i-1,j)=b(n,j);
            a(2*i,j)=b(m,j);
        end
    end

    %%变异
    for i=1:num
        for j=1:len
            y=rand();
            if y<p0
                a(i,j)=1-a(i,j);
            end
        end
    end

end

%%挑出适应度最大的样本
max=0;
k=0;
for i=1:num
    if fit(i)>max
        max=fit(i);
```

```
            k=i;
        end
    end

    %%输出
    for j=1:len
        fprintf(c,'%d ',a(k,j));
    end
    fprintf(c,'按此方法取点的平均成本为：%d',1/fit(k));
    fit(k)=fitness(1,k,a(k,:),cx,cy);

    %%关闭文件
    fclose(c);
```