# 统计推断在数模转换系统中的应用

第46 组　李佳荣5130379001, 王泽鹏5130309516

**摘要：** 本文阐述了如何使用插值、启发式搜索为传感器的非线性输入输出特性设计定标方案。该方案旨在确定在数据点较多情况下如何选取适当数量、适当位置的数据点对输入输出特性校准，并保证校准结果的准确性和校准方案的易操作性。在设计方案的过程中，我们主要利用遗传算法提高搜索效率，迭代时通过多项式拟合对单点定标误差成本进行估算，并根据采样点的数目设定单点测定成本，最终通过罚函数法进行筛选。根据一组实际的测量数据，我们得出了针对该组数据的一套定标方案，证明了所设计的校准方法的可行性和合理性。

**关键词：** 插值，启发式搜索，定标，遗传算法

# Application of Statistical Inference in DA Inverting System

Group 46　Li Jiarong 5130379001, Wang Zepeng 5130309516

**Abstract:** This article has discussed the method of calibrating the sensor’s nonlinear input and output characteristics using interpolation and heuristic search. This method aims to determine how to choose proper numbers and positions of data points to finish the task while the database is very huge, and make sure the accuracy of the result and the handleability of the method. In the process of design, we mainly used Genetic Algorithm to raise search efficiency, polynomial fitting while iterating to estimate single point calibration estimate error cost, and set a single point calibration cost according to numbers of sampling points, finally filtered data by penalty function method. We successfully made it eventually in line with the actual measurement data and proved the feasibility and rationality of our calibration method.

**Key words:** interpolation, heuristic search; calibration; Genetic Algorithm

# 1 引言

在该产品的生产中我们需要对该产品某模块的输入输出特性进行测量和定标。在定标中，选取的点越多，就越能符合该模块儿输入输出特性的实际情况。但问题在于，在实际生产中，如果选取过多的点定标，所花费的人力、时间等成本也将过高。因此，我们的目的在于如何选取适当的点可以较准确地完成定标工序又不至于定标成本过高。在各种搜索最优解的算法中，我们选择了遗传算法。他通过模拟自然选择的方式完成对最优解的搜索，优点在于他从问题解的串群开始搜索，而不是单个初始解，减少了陷入局部最优解的可能性。[1] 其中适应度函数的选择是该算法的核心之一，不同的函数对结果和拟合曲线的相似度都有不同的影响。

# 2 符号约定

表 1 符号约定

| | |
|---|---|
| **X** | 传感部件的输出电压 |
| **Y** | 传感部件的输入，即被监测物理量 |
| $\hat{\mathbf{Y}}$ | 监测模块的输出，即监测模块将传感部件输出 **X** 所转换成的读数（对 **Y** 的估测值） |
| $y_{i,j}$ | 第 $i$ 个样本之第 $j$ 点对应的 **Y** 实测值（来自标准样本数据库） |
| $\hat{y}_{i,j}$ | 第 $i$ 个样本之第 $j$ 点对应的 **Y** 估测值 |
| q | 单点测定成本 |
| $n_i$ | 对样本 $i$ 定标过程中的单点测定次数 |
| $\mathbf{S}_i$ | 对样本 $i$ 的定标成本 |
| **M** | 对样本 i 的定标成本 |
| **C** | 基于标准样本数据库评价一个校准方案，算得的该方案总体成本 |

# 3 主要数值计算方法

## 3.1 拟合

实现由给定点估算其他点的过程，可以根据这些数据生成一个函数表达式，再求表达式在其他各点的值作为该点的估计值，即曲线拟合。

本题中可以采用曲线拟合的方法。在 MATLAB 中，提供了多种拟合命令，这里我们使用多项式拟合命令 polyfit(xdata,ydata,n)。polyfit 函数的数学基础是最小二乘法拟合原理。它通过最小化误差的平方和寻找数据的最佳函数匹配。利用最小二乘法可以简便地求得未知的数据，并使得这些求得的数据与实际数据之间误差的平方和为最小。

## 3.2 插值

在该问题中，我们需要通过选定的部分点来推算其他点的估计值，而这种运用一组已知的点求出未知点的函数值的过程就是插值。他要求在离散数据的基础上补插连续函数，使得这条连续曲

线通过全部给定的离散数据点。插值是离散函数逼近的重要方法，利用它可通过函数在有限个点处的取值状况，估算出函数在其他点处的近似值。主要的插值方法有 Lagrange 插值与 Hermite 插值，其中后者不仅需要在给定离散点的函数值相同，更要求在这些节点上的导数也相同。

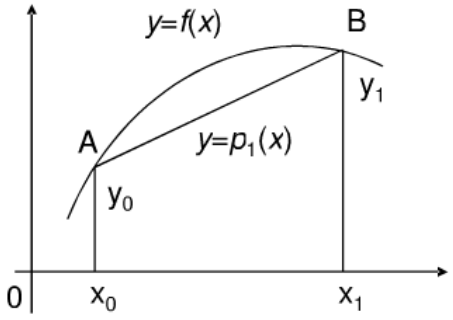



**图 1　二阶 Lagrange 插值（线性插值）**

### 3.2.1　分段三次 Hermite 插值

由于多项式插值会出现 Runge 现象——即随节点个数 $n$ 的增加，误差 $|R_n(x)|$ 不但没减小，反而不断的增大，人们逐渐用分段插值代替了原先的整体插值。分段插值即按节点分段，每一段内均用 $m$ 次多项式估计。而三次 Hermite 插值除要求用三次多项式插值外，还要求在节点上与被插值函数有相同的导数值，因此最终拟合结果是光滑的曲线。

### 3.2.2　三次样条插值

样条插值也是一种分段插值。所谓样条函数，即由一些按照某种光滑条件分段拼接起来的多项式组成的函数。早期工程师制图时，把富有弹性的细长木条（所谓样条）用压铁固定在样点上，在其他地方让它自由弯曲，然后沿木条画下曲线。成为样条曲线。

三次样条函数是最常用的样条函数，即每段函数均为三次函数。三次样条函数满足处处有二阶连续导数。图 2给出了 $y=\sin x$ 的示例，$x$ 取值范围 $[0,10]$，步长为 1.

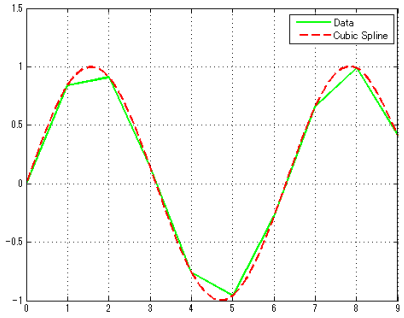



**图 2　$y=\sin x$ 三次样条插值**

### 3.2.3　分段 Hermite 插值与样条插值比较

分段 Hermite 插值与样条插值同为分段插值，都能一定程度上避免 Runge 现象，在拟合结果上有相似性，但两者还是有一些不同：

- 三次样条插值只要求在每个分段区间内用三次多项式拟合，而 Hermite 插值还要求拟合函数在节点的导数与原始值相同，这样做出的曲线更加光滑。

- 三次样条插值函数自身光滑，不需要知道 $f$ 内部各点的导数值（但可能仍需要两端点处导数值），他先由函数值确定导数值，再由三弯矩等方法解决问题；而 Hermite 插值依赖于 $f$ 在所有插值点的导数值，而这个导数值有时并不方便求。

我们将分别运用这两种插值方法对本题进行求解，并进行比较。

## 3.3 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。其本质是一种高效、并行、全局搜索的方法，它能在搜索过程中自动获取和积累有关搜索空间的知识，并自适应的控制搜索过程以求得最优解，就如同自然界中的优胜劣汰，适者生存。

遗传算法的主要过程可分为个体编码、初始化、适应度计算、选择、交叉、变异，并一代代的进行循环。[2] 如图 3所示为其主要流程图。

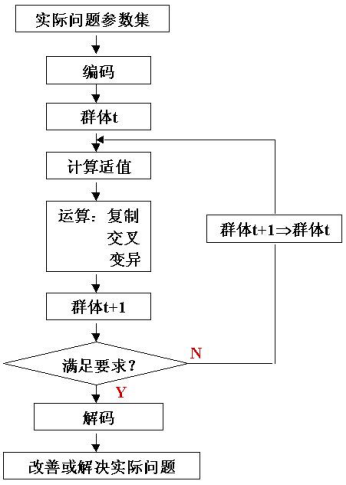



图 3　遗传算法流程图

(1) 个体编码

遗传算法的对象是表示个体的字符串，如果染色体与基因，所以需要把各个数据点表示为统一的字符串，本题采取无符号二进制整数表示。

(2) 初始化

我们需要确定种群规模，然后随机产生初始群体。

(3) 适应度计算

评价每个个体优劣的度，从而决定其遗传机会的大小。其核心是适应度函数，这也是整个遗传算法的关键。适应度函数的选择与具体的问题环境有关，对同一个题，选取不同的适应度函数，最后得到的结果也会不同。在本题中，我们以定标总成本作为其适应度的评价方式。

(4) 选择

选择是从群体中选择优胜个体、淘汰劣质个体的过程。选择常用的方法有适应度比例方法、随机遍历抽样法、局部选择法等。其中最常见的是适应度比例方法，也即轮盘赌选择法。每个个体的生存概率与其适应度成正比，然后通过产生 [0,1] 之间的均匀随机数来选出能产生后代的优质个体。

(5) 交叉

这是产生新个体的主要过程，他以某一概率和方法交换个体间的部分染色体。这里我们采用单点交叉的方法，即对群体随机配对，再随机选择交叉点的位置，最后进行染色体交换。

(6) **变异**

变异运算是对个体的某一个或某一些基因座上的基因值按某一较小的概率进行改变，它也是产生新个体的一种操作方法。对于二进制数来说，可以简单地用随机选取变异点，对应值取反来完成。

经过以上步骤后，我们就产生了下一代新的种群，如此循环下去，直到预先设定的遗传代数，或者种群优化程度已不再上升时停止。这样便可以得到最终结果。

# 4 传感部件输入输出特性定标方案设计

## 4.1 设计输入输出函数及采样点个数

选取给定数据中的前几组数据，画出 $\boldsymbol{Y}-\boldsymbol{X}$ 图像，根据初步观察，$\boldsymbol{Y}-\boldsymbol{X}$ 基本符合三次函数关系，故选用三次函数模型对 $\boldsymbol{Y}$—$\boldsymbol{X}$ 进行定标。确定一个三次函数的参数需要至少 4 个采样点，而根据经验，9 个采样点已经足以拟合出误差较小的三次多项式，故决定在程序中将采样点个数从 4 到 9 进行枚举，得到各确定采样点个数下的最小成本方案后优中取优。根据程序运行结果，最终选取 5 个采样点。

## 4.2 搜索最优方案

从 51 个样本点中选取 9 个数据点共有 $\mathrm{C}_{51}^{9}$ 约为 30 亿种可能，这使得穷举搜索最优方案几乎是不可能的，所以我们决定采用启发式搜索的方法来提高搜索效率。而遗传算法作为最经典的启发式搜索算法之一，具有通用性，自组织性和自学习性。故此处采用遗传算法完成进行启发式搜索。

### 4.2.1 遗传算法的参数确定

我们这里选定种群大小为 50，遗传代数为 100，代沟为 0.7，变异概率为 0.2。为防止种群早熟，我们决定采用较大规模的种群，较多的遗传代数，以及较大的变异概率。

### 4.2.2 适应度函数的设计

本着拟合函数准确性越高越好以及采样点越少越好的原则，我们设计了如下成本计算规则。

(1) **单点定标误差成本**

$$\mathbf{s}_{i,j}=\begin{cases}0 & \text{if } |\hat{y}_{i,j}-y_{i,j}|\leqslant 0.5\\ 0.5 & \text{if } 0.5<|\hat{y}_{i,j}-y_{i,j}|\leqslant 1\\ 1.5 & \text{if } 1<|\hat{y}_{i,j}-y_{i,j}|\leqslant 2\\ 6 & \text{if } 2<|\hat{y}_{i,j}-y_{i,j}|\leqslant 3\\ 12 & \text{if } 3<|\hat{y}_{i,j}-y_{i,j}|\leqslant 5\\ 25 & \text{if } |\hat{y}_{i,j}-y_{i,j}|>5\end{cases} \tag{1}$$

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 $\mathbf{Y}$ 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 $\mathbf{s}_{i,j}$ 记。

(2) **单点测定成本**

实施一次单点测定的成本以符号 q 记。这里指定 $\mathrm{q}=12$。

(3) 某一样本个体的定标成本

$$\mathbf{S}_i = \sum_{j=1}^{51} \mathbf{s}_{i,j} + \mathrm{q} \cdot \mathrm{n}_i \tag{2}$$

对样本 $i$ 总的定标成本按式 (2) 计算，式中 $\mathrm{n}_i$ 表示对该样本个体定标过程中的单点测定次数。

(4) 校准方案总体成本

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。

$$\mathbf{C} = \frac{1}{\mathbf{M}} \sum_{i=1}^{\mathbf{M}} \mathbf{S}_i \tag{3}$$

总体成本较低的校准方案，认定为较优方案。

### 4.2.3 确定校准方案

在完成上述步骤后，我们确定大致了校准方案。首先枚举采样点个数，在某个确定的采样点个数下，随机生成初始种群，然后根据其成本函数值进行筛选，筛选后进行交叉、变异等操作，直至遗传代数达到指定值，此时从当前种群中找到成本最低的一个个体，即为当前采样点个数下的最优校准方案。最后将各校准方案进行比较，最小成本的方案即为最终校准方案。

## 4.3 对方案的评价和分析

### 4.3.1 不同采样点的比较

根据我们的程序结果，不同采样点数的定标方案的最小成本比较见表2。

表 2 不同采样点的最小成本

| 采样点数 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|
| 成本 | 96.40 | 83.26 | 85.12 | 91.50 | 101.22 | 111.62 |

用 MATLAB 绘制统计图如图 4、5所示。对于同一张图，左上图，右上图，左下图是只和最终定标方案的关于误差、成本的统计结果相关。左上图内点 (x，y) 表示成本为 x 的数据点组数为 y。

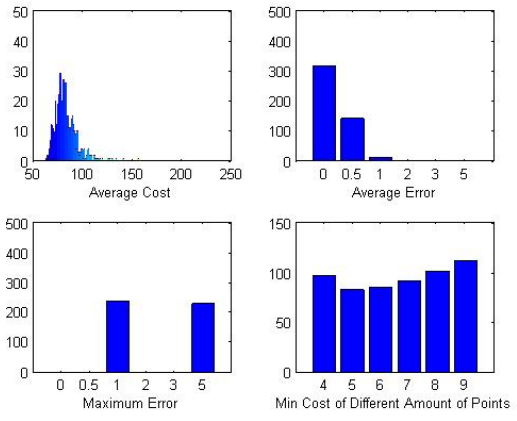



图 4 5 个采样点结果统计

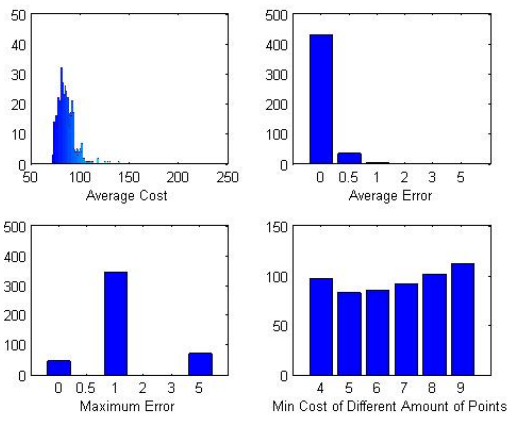


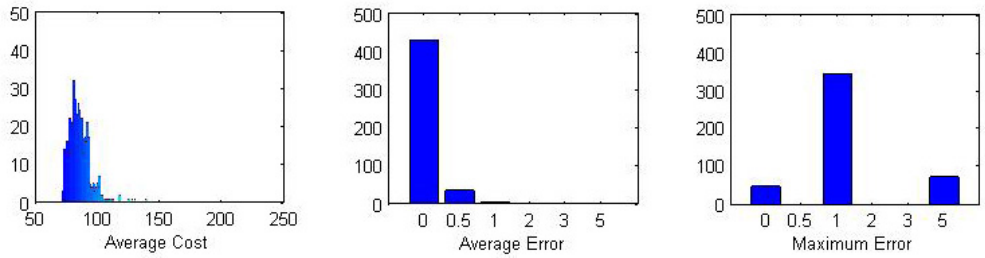

图 5 6 个采样点结果统计

右上图内点 (x，y) 表示平均误差（即不包含测定成本），其中横坐标表示误差落在前一个横坐标的值到该值的区间内。右下图是对不同定标方案的最小成本的比较。

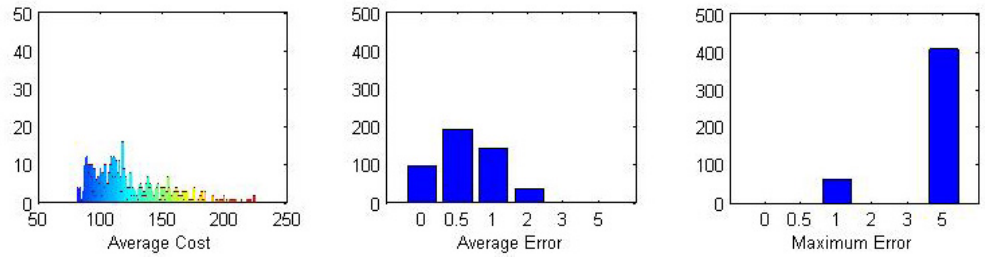
观察图 4、5可知，总成本最小方案是 5 个采样点（称为方案 A）。但是如果仅考虑单点测定误差的话，6 个采样点的方案（称为方案 B）每组平均误差为 1.12，5 个采样点的每组误差平均误差为 22.77，显然远大于方案 B。因为考虑到由于技术进步等原因，单点测定成本会随时间增长逐渐减小，所以我们认为方案 B 更适合应用到生产生活中。

### 4.3.2 不同拟合方法的比较

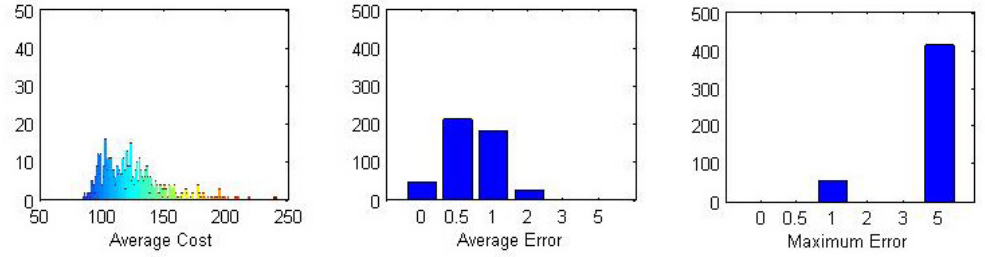
在程序运行的过程中，我们发现多项式拟合、三次样条插值、三次多项式插值的误差依次下降。这里我们以方案 B 为例，分别运行程序绘制出误差统计曲线，图 6中从上到下分别是对方案 B 三次多项式插值、三次样条插值，多项式拟合的结果。




(a) 分段三次多项式插值




(b) 三次样条插值




(c) 多项式拟合

图 6　不同拟合方法比较

从图中可明显看出，三次样条插值和多项式插值无论是从每组平均误差和每组最大误差上都远大于三次多项式插值。

## 5　关于算法的优化及其他思考

### 5.1　优化方法

为防止每一代中最优个体在变异、选择过程中被淘汰，我们这里采用如下“保留最优个体”的优化方法：用中间变量记录每一代的最优个体，在新一代产生后用其替代新一代中的一个非最优个体（我们这里默认替代个体 1，如果个体 1 为最优个体，则替代个体 2）。采用这样的优化方法后可以保证在遗传算法进行过程中，最小成本随代数增长非严格单调递减，加快收敛速度。

### 5.2 关于其他方案的一些思考

在本次研究过程中，笔者也了解到一些其他的寻找定标方案的算法，笔者在这里提出自己的一些意见和看法。

#### 5.2.1 对于“将遗传代数调整至 500 代”的思考

笔者认为，“遗传代数增大”这种优化有效的前提是种群不“早熟”，遗传代数盲目增长可能会导致程序运行效率低下，做许多无用功。而提高变异概率可能可以在一定程度上克服“早熟”，使得“增加遗传代数”这种优化方法更有效。

#### 5.2.2 对于“种群内不设定各个体维数”的思考

笔者认为，遗传算法的种群不设定个体维数（即同一种群内既有 6 采样点的方案，也有 7，8 采样点的方案），这样会造成遗传算法效率低下，遗传代数必须增加到足够大才有把握搜索出最优解。不过好处是不需要外重循环枚举采样点数即可自动搜索出最优方案。

## 6 结论

由以上模型和算法我们最终保留两个定标方案：

**方案 A** 采样点数为5 个，(4,16,26,35,48)，总成本为82.77。

**方案 B** 采样点数为6 个，(4,15,24,31,39,49)，总成本为85.12。

尽管方案 A 在本文所采用的适应度函数值下总定标成本更低，但我们认为方案 B 更具有普遍价值和意义。

## 7 下一步工作

由于时间人力等限制，本次研究成果尚有许多不足之处。如果有可能展开后续研究，我们希望：

(1) 对数据点进行分段拟合或插值，以得到更为精确的结果。
(2) 尝试不同的函数模型对数据点进行拟合或者插值。
(3) 用遗传算法结合模拟退火方法，弥补遗传算法收敛速度慢的缺点。或者尝试采用蚁群算法，粒子群算法等其他启发式搜索算法，以达到更快找到最优解或找到更优解的目的。

## 参考文献

[1] 席裕庚 and 柴天佑. 遗传算法综述. 控制理论与应用, 13(6):697--708, 1996.

[2] 段玉倩 and 贺家李. 遗传算法及其改进. 电力系统及其自动化学报, 10(1):39--52, 1998.

[3] 统计推断课程组. 统计推断在数模转换系统中的应用. 上海交通大学出版社, 2014.

# 8 附录

代码 1 遗传算法

```
A = csvread('20141010dataform.csv');

%设定参数
Nind = 50;
MAXGEN = 100;
GGAP = 0.7;
min_cost = 1000;
min_sche = zeros;
min_points = 4;

%min_cost_table用于记录各采样点下最小成本
min_cost_table = zeros(6,1);

%外重循环枚举采样点个数
for num_of_ans = 4:9
    BaseV = crtbase(num_of_ans,50);
    [Chrom,Lind,BaseV] = crtbp(Nind,BaseV);
    l = length(Chrom(:,1));
    for i = 1:l
        Chrom(i,:) = sort(Chrom(i,:));
    end
    ObjV = FitnessFunc(A,Chrom);
    tmpbest = Chrom(1,:);
    tmpmincost = ObjV(1,1);
    gen = 0;
    ObjV = ObjV';
    while gen < MAXGEN
        FitnV = ranking(ObjV);
        SelCh = select('sus',Chrom,FitnV,GGAP);
        SelCh = recombin('xovsp',SelCh,0.7);
        SelCh = mut(SelCh,0.3,BaseV);
        ObjVSel = FitnessFunc(A,SelCh);
        ObjVSel = ObjVSel';
        [Chrom,ObjV] = reins(Chrom,SelCh,1,1,ObjV,ObjVSel);
        [Y,I] = min(ObjV);
        %用上一代最优解替换当前这一代非最优解的个体
        if I ≠ 1
            Chrom(1,:) = tmpbest;
            ObjV(1,:) = tmpmincost;
        end
        if I == 1
            Chrom(2,:) = tmpbest;
            ObjV(2,:) = tmpmincost;
        end
        gen = gen + 1;

        fprintf('====I am the gorgeous split line.====Num of Points: ...
            %d====\n',num_of_ans);
        fprintf('Generation:  %d\n',gen);
        fprintf('Min cost   :  %5.2f\n',min(ObjV));

        %记录这一代的最优解，保留之后用于下一代的替换
        [Y,I] = min(ObjV);
```

```
        tmpbest = Chrom(I,:);
        tmpmincost = Y;
    end
    [Y,I] = min(ObjV);
    sche = Chrom(I,:) + 1;
    if Y < min_cost
        min_sche = sche;
        min_cost = Y;
        min_points = num_of_ans;
    end
    min_cost_table(num_of_ans - 3,1) = Y;
    fprintf('The final scheme is(%d points):\n',num_of_ans);
    disp(sche);
end
fprintf('The final scheme is:\n');
disp(min_sche);
fprintf('The final cost is  :%5.2f\n',min_cost);
```

代码 2　适应度函数

```
%适应度函数
function Fitvalue = FitnessFunc(A,Chrom)
size_of_chrom = length(Chrom(:,1));
num_of_points = length(Chrom(1,:));
Fitvalue = zeros(1,size_of_chrom);
for k = 1:size_of_chrom
    for i = 1:469
        xx = A(i * 2 - 1,:);
        yy = A(i * 2 ,:);
        xdata = zeros(1,num_of_points);
        ydata = zeros(1,num_of_points);
        sort(Chrom(k,:));
        b = unique(Chrom(k,:));
        if length(b) < num_of_points
            Fitvalue(1,k) = 469000;
            break;
        end
        for j = 1:num_of_points
            xdata(1,j) = xx(1,Chrom(k,j) + 1);
            ydata(1,j) = yy(1,Chrom(k,j) + 1);
        end
        %采用三次多项式插值
        estiydata = interp1(xdata,ydata,xx,'cubic');
        for j = 1:51
            if abs(estiydata(j) - yy(j)) ≤ 0.5
                estiydata(j) = 0;
                continue;
            end;
            if abs(estiydata(j) - yy(j)) ≤ 1
                estiydata(j) = 0.5;
                continue;
            end;
            if abs(estiydata(j) - yy(j)) ≤ 2
                estiydata(j) = 1.5;
                continue;
            end
            if abs(estiydata(j) - yy(j)) ≤ 3
```

```
                estiydata(j) = 6;
                continue;
            end
            if abs(estiydata(j) - yy(j)) ≤ 5
                estiydata(j) = 12;
                continue;
            end
            if abs(estiydata(j) - yy(j)) > 5
                estiydata(j) = 25;
                continue;
            end
        end
        Fitvalue(1,k) = Fitvalue(1,k) + sum(estiydata) + 12 * ...
            num_of_points;
    end
    Fitvalue(1,k) = Fitvalue(1,k) / 469;
end
end
```

**代码 3　绘图外部调用函数**

```
%用于调用draw_stat绘图函数，这里参数设定取决于ga4.m的运行结果
A = csvread('20141010dataform.csv');
sche = [4 15 24 31 39 49];
min_cost_table = [96.40;82.77;85.12;91.50;101.22;111.62];
draw_stat(sche,A,min_cost_table);
```

**代码 4　绘图函数**

```
%绘图函数，sche为需要计算误差的方案，A为数据库矩阵，
%min_cost_table为6*1矩阵，保存4--9采样点下最小成本
function val = draw_stat(sche,A,min_cost_table);
    points = length(sche);
    cost_of_every_group = zeros(1,200);
    max_error = zeros(1,6);
    average_error = zeros(1,6);
    for i = 1:469
        xx = A(i * 2 - 1,:);
        yy = A(i * 2 ,:);
        xdata = zeros(1,points);
        ydata = zeros(1,points);
        for j = 1:points
            xdata(1,j) = xx(1,sche(1,j) + 1);
            ydata(1,j) = yy(1,sche(1,j) + 1);
        end
        %根据绘制图像的需求，可以更改这里的语句，
        %如果采用注释掉的两行代码可以绘制多项式拟合下的误差统计图
        estiydata = interp1(xdata,ydata,xx,'cubic');
        %p = polyfit(xdata,ydata,3);
        %estiydata = polyval(p,xx);
        for j = 1:51
            if abs(estiydata(j) - yy(j)) ≤ 0.5
                estiydata(j) = 0;
                continue;
            end;
            if abs(estiydata(j) - yy(j)) ≤ 1
                estiydata(j) = 0.5;
```

```
                continue;
            end;
            if abs(estiydata(j) - yy(j)) ≤ 2
                estiydata(j) = 1.5;
                continue;
            end
            if abs(estiydata(j) - yy(j)) ≤ 3
                estiydata(j) = 6;
                continue;
            end
            if abs(estiydata(j) - yy(j)) ≤ 5
                estiydata(j) = 12;
                continue;
            end
            if abs(estiydata(j) - yy(j)) > 5
                estiydata(j) = 25;
                continue;
            end
        end
        error = sum(estiydata) / 51;
        max1 = max(estiydata);
        cost = sum(estiydata) + 12 * points;
        cost = floor(cost);
        %如果误差超过250认为统计意义不大，直接设定为250
        if (cost > 250)
            cost = 250;
        end;
        cost_of_every_group(1,cost - 50) = ...
           cost_of_every_group(1,cost - 50) + 1;
        if error ≤ 0.5
            average_error(1,1) = average_error(1,1) + 1;
        end;
        if error > 0.5 && error ≤ 1
            average_error(1,2) = average_error(1,2) + 1;
        end;
        if error > 1 && error ≤ 2
            average_error(1,3) = average_error(1,3)+ 1;
        end;
        if error > 2 && error ≤ 3
            average_error(1,4) = average_error(1,4) + 1;
        end;
        if error > 3 && error ≤ 5
            average_error(1,5) = average_error(1,5) + 1;
        end;
        if error > 5
            average_error(1,6) = average_error(1,6) + 1;
        end;
        if max1 ≤ 0.5
            max_error(1,1) = max_error(1,1) + 1;
        end;
        if max1 > 0.5 && max1 ≤ 1
            max_error(1,2) = max_error(1,2) + 1;
        end;
        if max1 > 1 && max1 ≤ 2
            max_error(1,3) = max_error(1,3) + 1;
        end;
        if max1 > 2 && max1 ≤ 3
            max_error(1,4) = max_error(1,4) + 1;
```

```
            end;
            if max1 > 3 && max1 ≤ 5
                max_error(1,5) = max_error(1,5) + 1;
            end;
            if max1 > 5
                max_error(1,6) = max_error(1,6) + 1;
            end;
        end
        %绘制四幅柱状图
        subplot(221);
        diaData1 = diag(cost_of_every_group);
        pic = bar(diaData1,'stacked');
        axis([0 200 0 50]);
        xlabel('Average Cost');
        set(gca,'XTicklabel',50:50:250);
        P = findobj(pic,'type','patch');
        for n = 1: length(P)
            set(P(n),'facecolor','b');
        end

        subplot(222);
        diaData2 = diag(average_error);
        pic = bar(diaData2,'stacked');
        axis([0 7 0 500]);
        xlabel('Average Error');
        set(gca,'XTicklabel',{'0','0.5','1','2','3','5'});
        P = findobj(pic,'type','patch');
        for n = 1: length(P)
            set(P(n),'facecolor','b');
        end

        subplot(223);
        diaData2 = diag(max_error);
        pic = bar(diaData2,'stacked');
        axis([0 7 0 500]);
        xlabel('Maximum Error');
        set(gca,'XTicklabel',{'0','0.5','1','2','3','5'});
        P = findobj(pic,'type','patch');
        for n = 1: length(P)
            set(P(n),'facecolor','b');
        end

        subplot(224);
        diaData2 = diag(min_cost_table);
        pic = bar(diaData2,'stacked');
        axis([0 7 0 150]);
        xlabel('Min Cost of Different Amount of Points');
        set(gca,'XTicklabel',{'4','5','6','7','8','9'});
        P = findobj(pic,'type','patch');
        for n = 1: length(P)
            set(P(n),'facecolor','b');
        end
        val = 0;
    end
```