参考声明：

统计推断课程，2015 年秋季学期第 59 组，成员王思涵，学号 5140309217, 李惠原，学号 5140309169, 在报告编写过程中，以下方面参考了往届报告，现列表说明：

主要参考项目	说明
代码方面（模拟退火部分）	《统计推断在数模模数转换中的应用》，王博远，2014 年秋季学期，组号 23 在该组报告附录提供的模拟退火程序代码基础上，进行了修改。
拟合方式描述方面	《统计推断在数模转换系统中的应用》，陈琦，2014 年秋季学期，组号 52 参考了该组报告的关于拟合方式的描述。

统计推断在数模转换系统中的应用

组号：59　　王思涵 5140309217　李惠原 5140309169

摘 要：本文以某型电子产品内部的检测模块为研究对象，为该产品寻求特性校准的方案。以降低校准成本并且提高精度为原则，本文介绍了如何利用 matlab 寻求通过几个点拟合出曲线的优化方法，并通过遗传算法及模拟退火算法进行启发式搜索，寻求最优的取点组合，解决一个组合优化问题。

关键词：多项式拟合, matlab，模拟退火算法，三次样条插值法，统计推断，遗传算法

Application of Statistical Inference in DA Inverting System

ABSTRACT: This article is to introduce an optimization method for the progress of sampling in engineering practice. Based on the principle of reducing cost of calibration and improving the accuracy, this article introduces how to find a better method to get the fitting function by several points with the help of the software Matlab. Apart from that, the article also illustrates how to use the principle of Genetic Algorithm and Simulated Annealing to locate the optimal combination of sampling points.

Key Words: polynomial fitting, matlab, Simulated annealing algorithm, cubic spline interpolation, Statistical Inference, Genetic Algorithm

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。

2. 模型

为了对本课题展开有效讨论，需建立一个数学模型，对问题的某些方面进行必要的描述和限定。

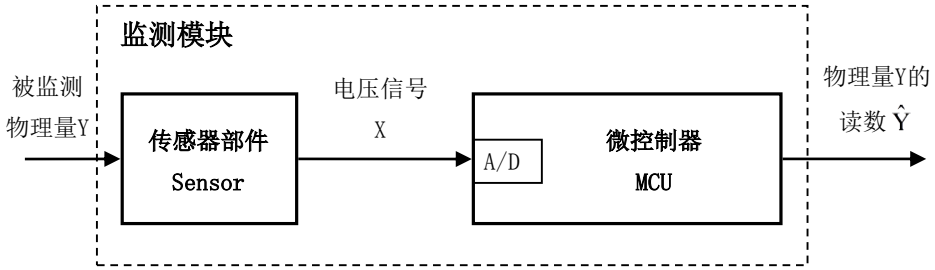


图 2-1 监测模块组成框图

监测模块的组成框图如图 1。其中，传感器部件（包含传感器元件及必要的放大电路、调理电路等）的特性是我们关注的重点。传感器部件监测的对象物理量以符号 Y 表示；传感部件的输出电压信号用符号 X 表示，该电压经模数转换器（ADC）成为数字编码，并能被微处理器程序所读取和处理，获得信号 $\hat{Y}$ 作为 Y 的读数（监测模块对 Y 的估测值）。

所谓传感特性校准，就是针对某一特定传感部件个体，通过有限次测定，估计其 Y 值与 X 值间一一对应的特性关系的过程。数学上可认为是确定适用于该个体的估测函数 $\hat{y}=f(x)$ 的过程，其中 x 是 X 的取值，$\hat{y}$ 是对应 Y 的估测值。

考虑实际工程中该监测模块的应用需求，同时为便于在本课题中开展讨论，我们将问题限于 X 为离散取值的情况，规定

$$X\in\{x_1,x_2,x_3,...,x_{50},x_{51}\}=\{5.0,5.1,5.2,...,9.9,10.0\}$$

相应的 Y 估测值记为 $\hat{y}_i=f(x_i)$，Y 实测值记为 y_i，$i=1,2,3,...,50,51$。

2.1 传感部件特性

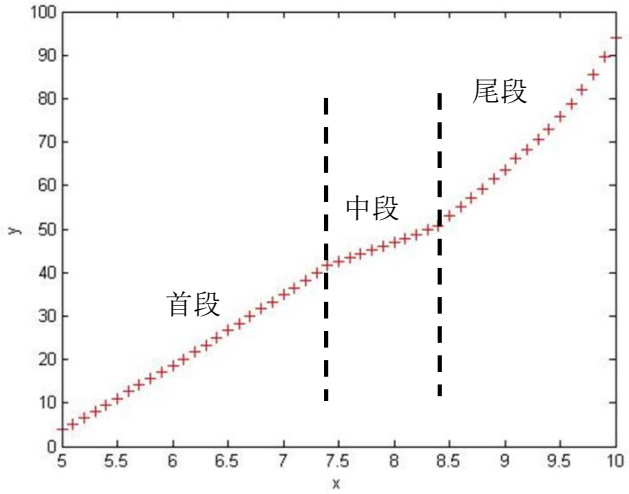


图 2-2 传感特性图示

一个传感部件个体的输入输出特性大致如图 2 所示，有以下主要特征：

- Y 取值随 X 取值的增大而单调递增；

- X 取值在[5.0,10.0]区间内，Y 取值在[0,100]区间内；
- 不同个体的特性曲线形态相似但两两相异；
- 特性曲线按斜率变化大致可以区分为首段、中段、尾段三部分，中段的平均斜率小于首段和尾段；
- 首段、中段、尾段单独都不是完全线性的，且不同个体的弯曲形态有随机性差异；
- 不同个体的中段起点位置、终点位置有随机性差异。

为进一步说明情况，图 3 对比展示了四个不同样品个体的特性曲线图示。

2. 2 标准样本数据库

前期已经通过试验性小批量生产，制造了一批传感部件样品，并通过实验测定了每个样品的特性数值。这可以作为本课题的统计学研究样本。数据被绘制成表格，称为本课题的“标准样本数据库”。

该表格以 CSV 格式制作为电子文件。表格中奇数行存放的取值，偶数行存放对应的取值。第 2i - 1 行存放第 i 个样本的 X 数值，第 2i 行相应列存放对应的实测 Y 数值。

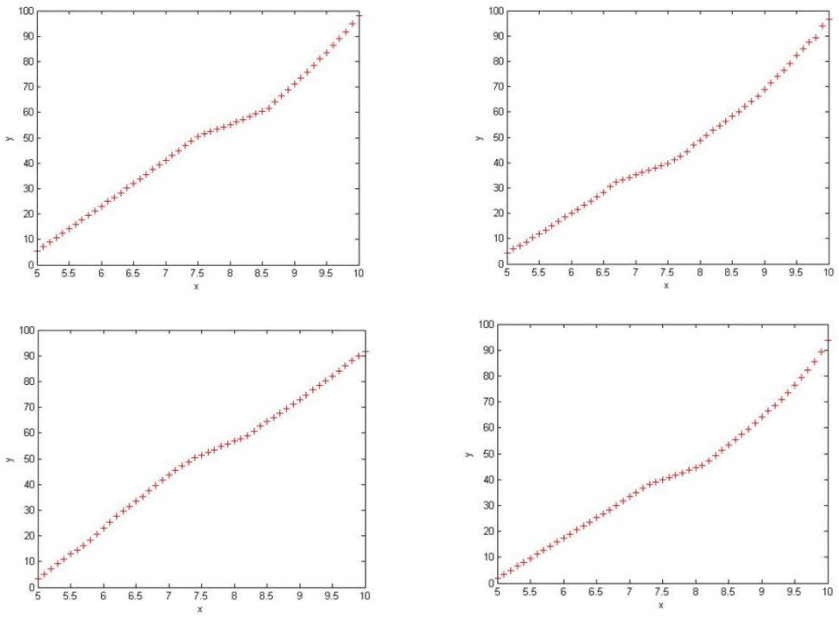


图 2-3 四个不同样本个体特性图示对比

2. 3 成本计算

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

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

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

$$S_i=\sum_{j=1}^{51}s_{i,j}+q\cdot n_i \quad (2\text{-}2)$$

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。

$$C=\frac{1}{M}\sum_{i=1}^{M}S_i \quad (2\text{-}3)$$

总成本较低的校准方案，认定为较优方案。

3. 选取拟合函数

选取拟合方式的过程中，我们探索利用何种函数进行拟合可以较为精确，并且成本相对较低地实现拟合。

3.1 多项式曲线拟合

假设一个三次方程为：

$$y=a_3x^3+a_2x^2+a_1x+a_0 \quad (3\text{-}1)$$

在每个样本的 51 个点中，我们选择七个点进行拟合，通过某一样本个体的定标成本

函数 $S_i = \sum_{j=1}^{51} s_{i,j} + q \cdot n_i$ 和校准方案总体成本函数 $C = \frac{1}{M}\sum_{i=1}^{M} S_i$ 计算得出平均成本。我们用MATLAB 写了拟合及计算成本的函数(见附录 1)。

选取 7 个点的过程中，基于上一届的成果（第 52 组报告），我们发现曲线大致分为三段，可以进行区间划分为[5,7.1],[7.2,7.7],[7.8,10]。为了保证选点具有代表性，拟合出的曲线相对准确，我们按照区间比例，在[5,7.1]区间均匀选取 3 个点，[7.2,7.7]均匀选取 1 个点，在[7.8,10]均匀选取 3 个点，进行曲线的拟合。

计算出的成本如下表所示：

所选择的 7 个点	平均成本
[5, 5.8, 6.6, 7.4, 8.2, 9, 9.8]	138.6405
[5, 5.7, 6.4, 7.5, 8.4, 9.1, 9.8]	140.8658
[5, 5.9, 6.8, 7.6, 8.5, 9.4, 10]	141.2655

表 3-1 多项式拟合平均

3.2 三次样条插值法拟合

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续。我们利用 MATLAB 中的插值函数 spline，我们用它写了插值及计算成本函数的代码（见附录 2）。通过取与多项式拟合方法相同的三组点，计算成本如下表所示：

所选择的 7 个点	平均成本
[5, 5.8, 6.6, 7.4, 8.2, 9, 9.8]	98.8185
[5, 5.7, 6.4, 7.5, 8.4, 9.1, 9.8]	101.6003
[5, 5.9, 6.8, 7.6, 8.5, 9.4, 10]	97.7265

表 3-2 三次样条插值平均成本

通过表 1 和表 2 的对比，发现用三次样条插值法计算得出的成本较低，所以我们决定用三次样条插值法得出拟合表达式。

4. 基于遗传算法及模拟退火算法寻找最优解

4.1 遗传算法

4.1.1 遗传算法介绍[4]

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。

遗传算法主要分为以下几个步骤。首先随机建立初始状态，即个体基因的第一代，根据实际问题求解函数来设定适应度函数。其次在接下来的若干代中，依次进行：通过每个个体适应度函数，由适应度的大小选择用赌轮法确定，个体间进行交叉和变异，产生有更好的适应度的下一代。

4.1.2 利用遗传算法求解的步骤[1,2]

（1）初始化(main)：设立初始种群 M 个个体，每个个体基因的长度是 51，每个基因序列中的基因点用 0 和 1 填充进行表示，0 和 1 两个状态表示选取或者是不选取该点。

（2）适应度函数(fitness)：适应度函数中进行了两个操作，即适应度的计算与为下一代的配对做准备。通过选定的拟合方式来计算每一个个体的定标成本，其中有误差成本和取点成本的两个部分。

（3）适应度选择(fitness)、交叉(selectCrossBest)及变异(selectMutationBest)：我们选择用倒数来评判适应度的选择，对种群中的每个个体拟合得到的成本取倒数，我们曾经尝试取平方来提高收敛速度，但是因为成本差不多，而且没有现有的成本小，最后就没选用平方取法。成本小的就是适应度高的，这样的方案应该被保留，反之，成本大的适应度相对就低，这样的方案应当被淘汰，再除以倒数之和，这样得到的用 0 到 1 的小数就是适应度。于是就可以求出每个个体对应的适应度，并且满足误差小的适应度高的条件。为了更好的方便配对操作，计算完适应度之后，增加了一个赌轮选择操作，通过赌轮操作选出过渡种群，即根据适应度优胜劣汰但不进行配对的部分。关于赌轮操作，对适应度矩阵，从第二项开始，进行累加，每一项和它的前一项构成一个区间，这个区间就作为一块赌轮轮盘上的区域分块。然后再通过随机数函数生成 M 个（种群大小）数据在 0~1 之间的随机数矩阵，将此 M 个数每个都与每一块赌轮轮盘上区域分块进行比较，如果落在分块里，就记录其对应的父代的位置。

（4）输出该代所有个体中适应度最小的一个，回到步骤（2），依次循环，直到设定的代数。

4.1.3 遗传算法数据分析及结论

（代码见附录 3）

取点方案	总成本

[2,11,19,27,34,45,50]	96.7095
[3,11,19,27,34,45,50]	96.7088
[3,10,19,27,34,45,50]	96.1413
[3,10,19,27,34 ,43,50]	95.5295
[3,9,19,27,34,43,50]	95.5023
[2,9,19,27,34,44,50]	95.2897
[2,9,20,27,34,44,50]	95.2147

表 4-1 遗传算法三次样条插值得到的最优解

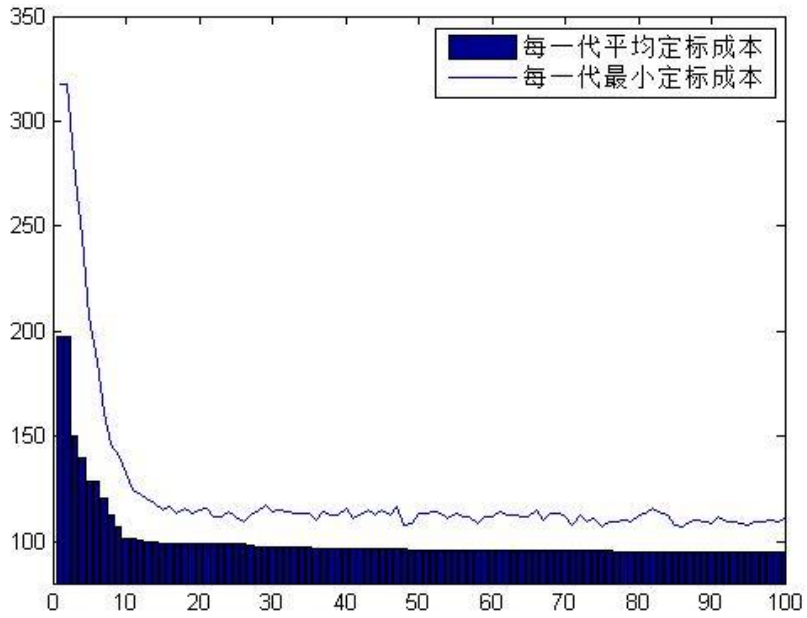


图 4-1 遗传算法成本图像

由表格及图像可知，在进行了 100 代之后，总成本最后基本稳定在 95.2147，故在选点方案为[2,9,20,27,34,44,50]时方案成本最小。

4.2 模拟退火算法

4.2.1 模拟退火算法介绍

模拟退火算法是基于 Monte-Carlo 迭代求解策略的一种随机寻优算法，其出发点是基于物理中固体物质的退火过程与一般组合优化问题之间的相似性。模拟退火算法从某一较高初温出发，伴随温度参数的不断下降,结合概率突跳特性在解空间中随机寻找目标函数的全局最优解，即在局部最优解能概率性地跳出并最终趋于全局最优。[3]

4.2.2 利用模拟退火算法求解的步骤

（1）初始化：仅设立一个个体，其基因长度为 51，每个基因序列中的基因点用 0 和 1 填充进行表示，0 和 1 两个状态表示选取或者是不选取该点。

（2）“能量”函数：由事先选定的拟合函数测定该取点方案下的误差成本和测定成本，相加得到定标成本。

（3）对于每一个降低到的温度，迭代若干次：求原解和新解的能量值，扰动求新解，比较原解和新解对应的“能量”值，如果新解“能量”值低，即成本低，则采取新解，或者以一定的小概率接受新解。

（4）若达到一定的次数没有接受新解则跳出算法，直接结束。否则迭代完次数后降低温度，回到步骤（3），直到降低到最小的温度为止。

4.2.3 模拟退火算法数据分析及结论

（代码见附录 4）

取点方案	总成本
[2,10,19,26,34,42,50]	96.4588
[2,10,20,26,34,42,50]	96.3673
[2,9,20,26,34,42,50]	96.2965
[2,9,20,26,33,42,50]	96.0860
[2,9,20,26,33,43,50]	95.1338

表 4-2 模拟退火算法三次样条插值得到的最优解

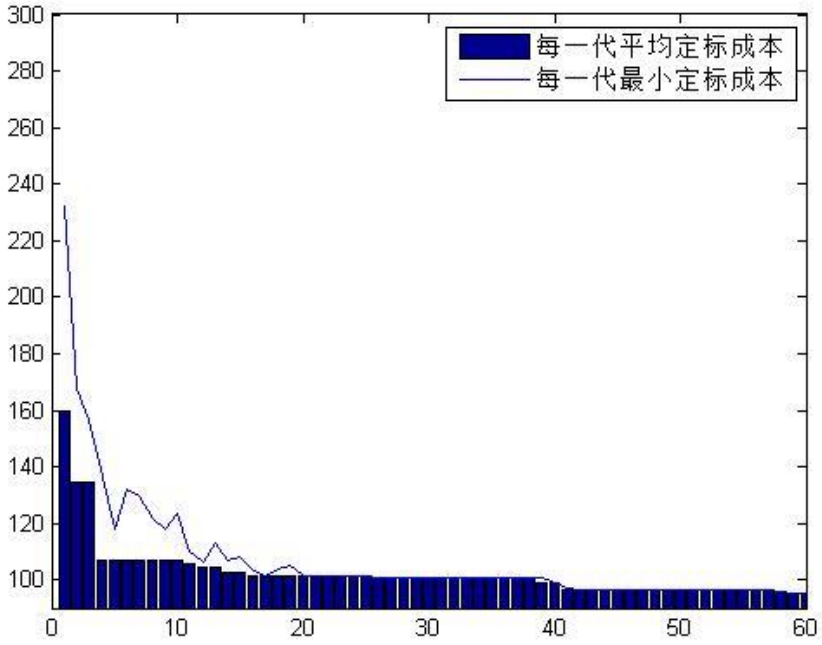


图 4-2 模拟退火算法成本图像

由程序运行结果可知，利用模拟退火算法得出最优组合解时取点数一直在 6 个和 7 个之间波动，然而一般状态是在 7 个点，只有极少数几次是 6 个点，而成本也是在 94.5-95.5 之间上下浮动。最终，我们组认为可以选择拟定最好的取点数为 7，最好的取点方案是 [2,9,20,26,33,43,50]，最小成本为 95.1338。

4.3 遗传算法与模拟退火算法的比较

4.3.1 结果比较

在遗传算法与模拟退火算法分别得到的几个最优解中，经过多次运行程序之后，可以看出，尽管两者计算出来的成本相差不多，但是模拟退火算出来的对应取点方案的成本小于遗传算法所得的成本的概率比反过来比较大的多，因此我们组认为模拟退火算法所得的取点组合方案相比遗传算法有更小的成本。

4.3.2 收敛比较

由图像可以直观地看出来，模拟退火算法在求最优解的过程中，最小成本存在上下跳动，反弹波动的情况，但从总体来看，模拟退火算法相比遗传算法的成本更小，最小成本的收敛速度更快。

4.4. 结论

通过遗传算法及模拟退火算法的多次尝试，我们得出最佳取点方案为取 7 个点，[2,9,20,26,33,43,50]，最小成本为 95.1338。

5．拓展部分：关于遗传算法变异参数对结果的影响

根据参考资料[5]，变异的参数一般设为 0.01~0.001，变异的参数增大，则突变的概率变大，有可能产生更优的解，但是同时也可能产生误差更大的个体。下面的两张图展示了遗传算法中，变异概率分别为 0.005 和 0.01 的求解最优成本过程。

（1）变异概率 Rate=0.005，种群数量为 200，利用遗传算法运行 60 代，得到结果如下:

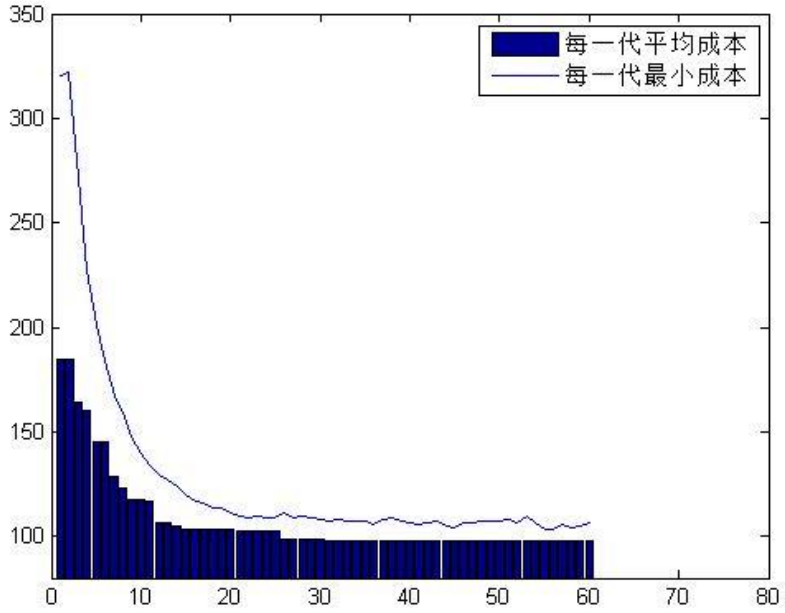
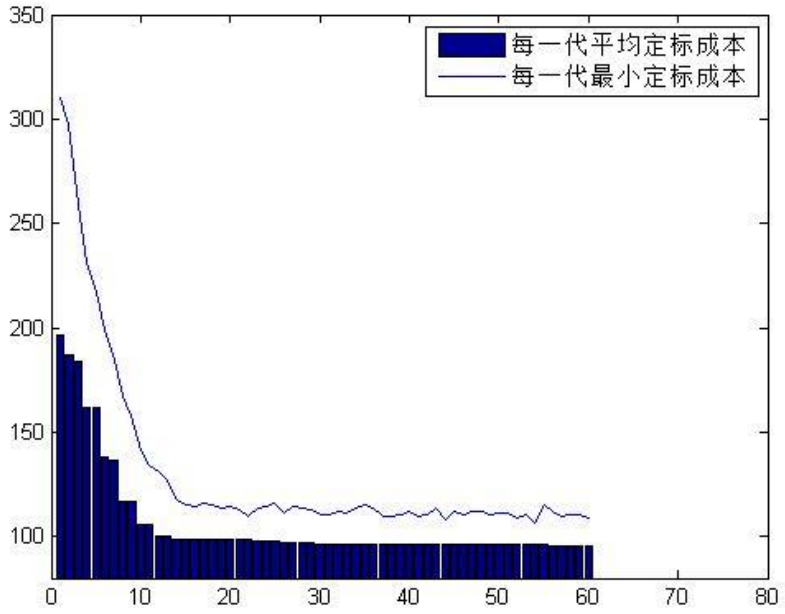


图 5-1 遗传算法变异概率对结果影响(Rate=0.005)

此时最优取点方案为[5,10,21,29,37,44,50]，最小成本为100.1840，平均成本为108.5984。

（2）变异概率 Rate=0.01，种群数量为 200，利用遗传算法运行 60 代，得到结果如下:



此时最优取点方案为[2,9,20,28,35,44,51]，最小成本为 95.4955，平均成本为 110.6057。

因此，我们组得出的结论是：从图中可以看出，变异概率变大，平均误差产生的突变程度就更大。

6. 参考文献

[1]卓金武 . MATLAB 在数学建模中的应用[M] . 北京：北京航空航天大学出版社，2003：79-120

[2]简单遗传算法 MATLAB 实现

http://www.cnblogs.com/biaoyu/archive/2011/12/03/2274604.html

[3]模拟退火算法_百度百科

http://baike.baidu.com/link?url=NtVZPJmpEGo0-luAoaVKe2q4lCi6jSdU5vtW0iIW0O-IpEfa-zBtDdNS24VnpeclBFeqTBv4_z3ul4QbCeNLu_

[4]遗传算法_百度百科

http://baike.baidu.com/link?url=U928my5Xamu7gQNgclFaqGB08rBtpSHseHPQSHPAIzmg1v83c2u92SuNGQzRkxXeCRUJlemNffMKxxb6aFUB-K

[5]遗传算法_百度文库

http://wenku.baidu.com/link?url=i8OdlD3RE-DSjWRKSlNG3xthGNsVac8-X4CEyncSMGuqnYhOhjD8GLRDHmOzW05sv2mPuOwJM2nNuo0dvAF4YLyjM0_gsdyQkN2A2ZXMKJa

[6]test_my_anwser 中成本函数：为防止自己成本函数出现错误，故而在检查自己函数无误后，直接选择老师给的成本函数计算方法。

附录 1 多项式拟合曲线并计算成本（原创代码）

```
%三次多项式拟合，计算成本
x=[5 5.8 6.6 7.4 8.2 9 9.8];            %用于计算拟合函数的x值
sum=0;                                  %总成本
for i=1:400                             %400个样本
    errorcap=0;                         %单个样品的误差成本

    y=[data(2*i,1) data(2*i,9) data(2*i,17) data(2*i,25) data(2*i,33)
data(2*i,41) data(2*i,49)];
    p=polyfit(x,y,3);
    a3=p(1);                            %存储多项式系数
    a2=p(2);
    a1=p(3);
    a0=p(4);
    for j=1:51
        %计算每个样品的误差成本
        xx=5.0+0.1*(j-1);
        yy=a3*xx*xx*xx+a2*xx*xx+a1*xx+a0;
        if(abs(yy-data(2*i,j))<=0.4)
           errorcap=errorcap+0;
        elseif(abs(yy-data(2*i,j))<=0.6)
               errorcap=errorcap+0.1;
               continue;
        elseif(abs(yy-data(2*i,j))<=0.8)
               errorcap=errorcap+0.7;
               continue;
        elseif(abs(yy-data(2*i,j))<=1)
               errorcap=errorcap+0.9;
               continue;
        elseif(abs(yy-data(2*i,j))<=2)
               errorcap=errorcap+1.5;
               continue;
        elseif(abs(yy-data(2*i,j))<=3)
               errorcap=errorcap+6;
               continue;
        elseif(abs(yy-data(2*i,j))<=5)
               errorcap=errorcap+12;
               continue;
        elseif(abs(yy-data(2*i,j))>5)
               errorcap=errorcap+25;
        end
    end
    sum=sum+errorcap+12*7;
end
```

```
sum=sum/400;
fprintf('%d',sum);
```

附录 2 三次样条插值法拟合曲线并计算成本（原创代码）

```
%三次样条插值法计算拟合函数，并计算成本
sum=0;  %平均总成本

x=[5 6.2 7.4 8.6 10];              %用于拟合的点横坐标
for i=1:400
    errorcap=0;                    %单个样本的误差成本
    y=[data(2*i,1) data(2*i,13) data(2*i,25) data(2*i,37) data(2*i,51)];
                                   %用于拟合的点的纵坐标
    xx=[5.0:0.1:10.0];             %x坐标，5-10，开始计算误差
    yy=spline(x,y,xx);             %插值得1*51矩阵yy表示插值后在5-10的取值
    for j=1:51                     %计算第i个样本第j个点的误差成本
        if(abs(yy(j)-data(2*i,j))<=0.4)
            errorcap=errorcap+0;
        elseif(abs(yy(j)-data(2*i,j))<=0.6)
                errorcap=errorcap+0.1;
                continue;
        elseif(abs(yy(j)-data(2*i,j))<=0.8)
                errorcap=errorcap+0.7;
                continue;
        elseif(abs(yy(j)-data(2*i,j))<=1)
                errorcap=errorcap+0.9;
                continue;
        elseif(abs(yy(j)-data(2*i,j))<=2)
                errorcap=errorcap+1.5;
                continue;
        elseif(abs(yy(j)-data(2*i,j))<=3)
                errorcap=errorcap+6;
                continue;
        elseif(abs(yy(j)-data(2*i,j))<=5)
                errorcap=errorcap+12;
                continue;
        elseif(abs(yy(j)-data(2*i,j))>5)
                errorcap=errorcap+25;
        end
    end
    sum=sum+5*12+errorcap;
end
sum=sum/400;
fprintf('%d',sum);
```

附录3 利用遗传算法寻找取点方案

1.main函数（主程序）

```
%主程序
tic;
clear all;clc;
fid = fopen('output.txt','wt');                %输出文件
minput=dlmread('20150915dataform.csv');    %从xlsx中读取数据
M0=800;N=51;
samplenum=M0/2;npoint=N;                       %设置样本点数数量和点数规模
global x;
x=zeros(1,npoint);
global originY;
global min_costChosenY;                        %记录每一代最小成本对应的取点方案
global min_cost;                               %记录每一代中最小的成本
global average;
originY=zeros(samplenum,npoint);
calculateY=zeros(400,51);                      %开辟了一个用于计算拟合结果的矩阵
M=200;                                         %设定种群数量
G=100;                                         %设定遗传代数
Result=zeros(1,M);
for i=1:samplenum
    x(1,:)=minput(i*2-1,:);
    originY(i,:)=minput(2*i,:);
    end                                        %把xlsx中的数据全部读入
                                               x,originY矩阵Recordmin_cost
                                               中

Recordmin_cost=[];
Recordaverage=[];
Gene=round(1*rand(M,51));                      %随机生成M行51列的0,1矩阵代表群
体基因初始化
for generation=1:G                                      %开始遗传算法
    Result = Cost( M,Result,Gene);                      %计算成本
    son = fitness( M,Result,Gene);                      %根据成本进行概率选择
    sonCross= selectCrossBest(son,M);                   %交叉
    sonMutate = selectMutationBest(sonCross,M);         %变异
    Gene=sonMutate;                                %更替新的基因作为下一代的基因
    generation
    min_costChosenY
    min_cost
    Recordmin_cost=[Recordmin_cost,min_cost];
    Recordaverage=[Recordaverage,average];
    fprintf(fid,'generation:%d \n ',generation);
    fprintf(fid,'min_cost:%.4f \n',min_cost);
    average
```

```
end
bar(Recordmin_cost)
axis([0 100 80 350]);
hold on;
plot(Recordaverage)
axis([0 100 80 350]);
toc;
```

2. fitness函数（计算种群适应度）

```
function son = fitness( num,m,parent)
%适应度函数，包括了适应度的计算和自然选择的过程
%保留最优的个体，num表示种群大小，m表示fitness的误差结果
m=1./m;                    %取适应度倒数
%m=sum1*m;
sum2=sum(m);              %计算矩阵的和以使适应度用小数表示，并且表示适应度的和为1
m=1/sum2*m;               %适应度矩阵
adp=zeros(1,num)+1;       %记录产生的过度种群中的个体在原来种群中的位置
son=zeros(num,51);        %产生过渡种群
b=m(1);                   %用于找最小的误差，一开始选取第一个个体，然后在逐个比较
e=1;                      %记录误差最小值也是适应度最小的个体的位置
for i=2:1:num             %对于每一个适应度进行累加，得到赌轮轮盘区域
    if(m(i)>b)            %用比较法寻找适应度最大的个体
        e=i;
        b=m(i);
    end
    m(i)=m(i-1)+m(i);                              %累加
end
for i=2:1:num             %从子代第二个开始记录，第一个为最优个体
    for j=1:1:num                  %与赌轮各个区域逐个比较
        c=rand;                                    %产生随机数
    if (c<m(j)&&j==1) adp(i)=j ;                   %落在第一个区域
    continue;
    end
    if (c<m(j)&&c>=m(j-1))adp(i)=j ;    %落在第j个区域，记录位置j，并跳出循环
    continue;
    end
    end
end
son(1,:)=parent(e,:);      %生成的过渡矩阵中的第一个个体为上一代的最优个体
 for i=2:1:num
    son(i,:)=parent(adp(i),:); %取出赌轮结果中对应的父代个体，构成过渡代
end
out=son;                                   %返回过渡代
end
```

3. cost函数（计算成本）

```
function Result = Cost( M,Result,Gene )
global originY;
global x;
global min_cost;                              %min_cost记录每代中最小的成本
min_cost=12*51+400*51*25;
global min_costChosenY;     %min_costChosenY记录每代个体中最小成本对应的取点方案
global average;
average=0;
for m=1:M
    [Xtemp,Ytemp]=find(Gene(m,:)~=0);
    count=length(Xtemp);
    tempChosenX=Ytemp.*0.1+4.9;
    ChosenY=originY(:,Ytemp);
    calculateY=spline(tempChosenX,ChosenY,x);      %三次样条插值拟合
    errorCount=abs(calculateY-originY);            %误差成本计算
    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;     %计算平均成本
    if Result(1,m)<min_cost                        %比较是否该代中最小成本
        min_costChosenx=tempChosenX;
        min_costChosenY=Ytemp;
        min_cost=Result(1,m);
    end
end
average=sum(Result)/M;
out=Result;
end
```

4. selectCrossBest函数（交叉后产生新父代）

```
function  m1=selectCrossBest(m1,num)
```

```
%选择交叉后父代子代中较优的作为新的父代
%过渡代的配对，产生新的一代
%num为种群大小，m为适应度产生的过渡矩阵
%采用生成num/2*1的随机数来决定交换基因的位置
k=num/2;                                    %两两配对，k为个数的一半
matrix=randi(51,k,1);                       %生成范围为1-51的随机数k*1矩阵
for i=1:1:k                                 %配对k次
    if (matrix(i)==51)                      %是否是最后一位，若是最后一位，没发交换，
后面的操作不继续
    continue
    end
    tmp1=[m1(i+1,1:matrix(i)),m1(num-i+1,(matrix(i)+1):51) ];
                                            %交换基因片段，产生新的个体
    tmp2=[m1(num-i+1,1:matrix(i)),m1(i+1,(matrix(i)+1):51) ];
    if (i~=1) m1(i,:)=tmp1;                 %若位置不是1，则交换新的基因;
    end
    m1(num-i+1,:)=tmp2;                     %新的个体的产生
end
out=m1;                                     %新的种群的产生
end
```

5.selectMutationBest函数（变异后产生新父代）

```
function m2=selectMutationBest( m2,num )
%选择变异之后父代子代中较优的个体作为新的父代
%变异函数，对于每个基因点0.01的概率变异
%m2为新的种群，num为种群大小
for i=1:51                                  %每个基因点
    for j=2:num                             %对于每个个体，第一个最优个体保留，不变异
        if (rand<0.01)                      %变异的概率选择的是0.01，此处可以更改数据
            m2(j,i) = 1-m2(j,i);            %0变1,1变0
        end
    end
end
```

附录 4 利用模拟退火算法寻找取点方案

1.main 函数（主程序）

```
tic;
clear all;
clc;
%fid = fopen('output2.txt','wt');
minput=dlmread('20150915dataform.csv');
M0=800;N=51;
samplenum=M0/2;npoint=N;
```

```
global x;
x=zeros(1,npoint);
global originY;
global tempMaxChosenY;
average=0;
originY=zeros(samplenum,npoint);
calculateY=zeros(400,51);
M=1;
Result1=zeros(1,1);
Result2=zeros(1,1);
for i=1:samplenum
    x(1,:)=minput(i*2-1,:);
    originY(i,:)=minput(2*i,:);          %从原始xlsx中读入数据
end
tempMin=51*12+51*25;
Gene1=round(1*rand(M,51));
T=100000;                                %初始设定的温度
generation=1;
Record1=[];Record2=[];Record3=[];
while T>0.1                              %直到退火达到的最小温度
    flag2=1;k=0;Record2=[];
    for t=1:90
        flag = 0;
        Gene2 = Mutation(Gene1);
        Result1 = cost1(1,Result1,Gene1);
        Record2=[Record2,sum(Result1)];
        Result2 = cost1( 1,Result2 ,Gene2);
        t=Result2-Result1;
        if t<0|rand<exp(-t/(T*0.0001))  %新解优于原解或以一定的小概率接受新解
            Gene1 = Gene2;
            flag=1;
        if Result2<tempMin
            tempMin=Result2;
            Recorder=tempMaxChosenY;
        end
        end
        if flag==1
            k = 0;
        else
            k = k+1;
        end
        if k>40
            flag2=0;
        end
```

```
        if k>40
           break;
        end
    end
    average=sum(Record2)/90;
    if flag2==0 break;
    end
    T
    T=T*0.9;
    tempMin
    Record1=[Record1,tempMin];        %Record1记录每一个温度中的最小定标值
    Record3=[Record3,average];        %Record3记录每一个温度的平均值
    Recorder
    generation
    generation=generation+1;
end
bar(Record1)                          %画出最小定标值和平均值的图形
axis([0 60 80 150]);
hold on;
plot(Record3)
axis([0 60 90 300]);
toc;
```

2. cost 函数（计算误差成本及测定成本）

```
global originY;
global x;
global tempMaxChosenY;
tempMax=12*51+400*51*25;
m=1;
[Xtemp,Ytemp]=find(Gene(m,:)~=0);
count=length(Xtemp);
tempChosenX=Ytemp.*0.1+4.9;
ChosenY=originY(:,Ytemp);
calculateY=spline(tempChosenX,ChosenY,x);      %采用三次样条插值方法
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
tempMaxChosenx=tempChosenX;
tempMaxChosenY=Ytemp;
tempMaxChosenY;
out=Result;
end
```

3.mutation 函数（变异）

```
function m = Mutation(m)
global num;
for i=1:51
    if (rand<0.03)
        m(1,i) = 1-m(1,i);
    end
end
out = m;
end
```