统计推断在数模转换系统中的应用

第 29 组 孙铭君 5140309374 罗天洋 5140309367

摘要：本文为上海交通大学电子信息与电气工程学院课程设计《统计推断在数模转换系统中的应用》的课题论文。本小组运用 Matlab 数学软件并结合统计推断的方法，通过对样本数据进行研究，使用遗传算法合理选择取点方案，最终计算并比较成本确定较优方案。

关键词： Matlab，三次多项式拟合，三次样条插值拟合，启发式搜索，遗传算法

1 引言

在工程时间和科学实验中，当我们需要寻找两个变量间的关系是，在理论上我们可以通过计算得出两个变量之间的确定的函数关系，但实际却很难找到非常精确描述两个变量的函数。通常采取的方法是通过多种不同曲线拟合，算出多种不同的方案，并选择某一统计量作为评定优劣标准，从而选出最优的方案。

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。

1.2 模型建立

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。

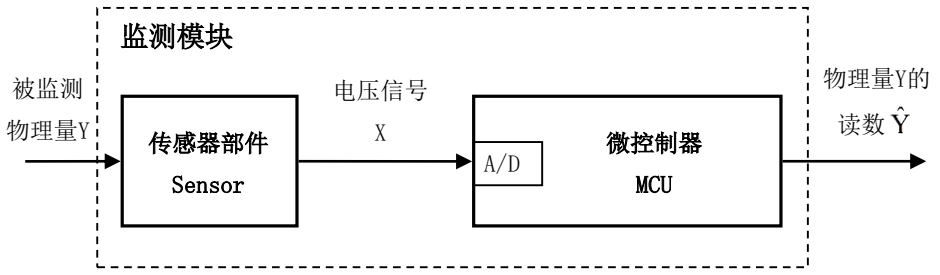

图 1 监测模块组成框图

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， $i = 1, 2, 3, ..., 50, 51$。

1.3 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。

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

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

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

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。

$$C = \frac{1}{M} \sum_{i=1}^{M} S_i \qquad (2)$$

总成本较低的校准方案，认定为较优方案。

1.4 传感器部件特性

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。

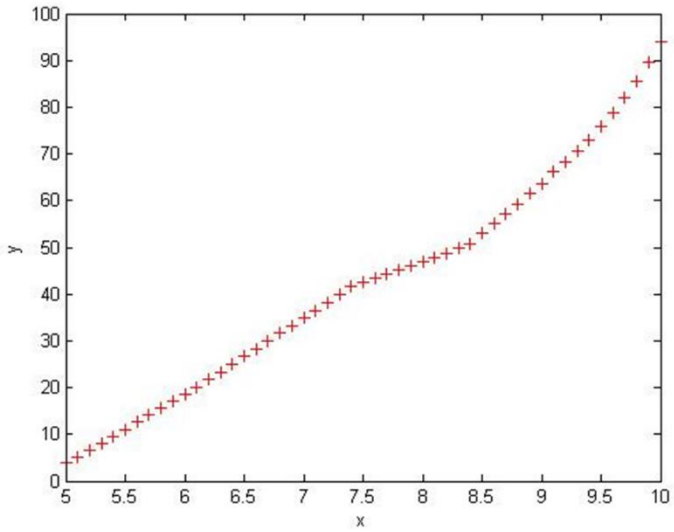

图 2 传感特性图示

2 拟合方法选取

适用于本课程的拟合方法有许多，常用的有多项式拟合、指数函数拟合、插值法拟合、高斯拟合、洛伦兹拟合和傅里叶级数拟合等。由样本数据特性可知以下特征：Y 取值随 X 取值的增大而单调递增，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，不同个体的中段起点位置、终点位置有随机性差异。我们选择使用多项式拟合与样条插值拟合进行比较。

2.1 多项式拟合

多项式拟合是一个比较常见的拟合方式，易于理解， 由数据看出大致分为单调递增的三段函数，与三次函数的大致图像很相似，所以采用三次多项式拟合进行尝试。通过 MATLAB 自带多项式拟合函数 polyfit 计算，拟合出三次多项式。由于函数大致分为三段，不严格符合三项式函数平滑特点，存在一定误差。

2.2 样条差值拟合

在数值分析这个数学分支中，样条插值是使用一种名为样条的特殊分段多项式进行插值的形式。由于样条插值可以使用低阶多项式样条实现较小的插值误差，这样就避免了使用高阶多项式所出现的龙格现象，所以样条插值得到了流行。样条插值拟合是数学分析中一种常用的拟合方式，本次采用三次样条插值进行拟合。三次样条函数:函数 $S(x)\in C2[a,b]$ ，且在每个小区间[xj,xj+1]上是三次多项式，其中 a =x0 <x1<...< xn= b 是给定节点，则称 S(x)是节点 x0,x1,...xn 上的三次样条函数。若在节点 x j 上给定函数值 Yj= f (Xj).(j =0, 1, , n) ，并成立 S(xj) =yj .(j= 0, 1, , n) ，则称 S(x)为三次样条插值函数。样条插值所得曲线是光滑曲线，且导数曲线光滑，在观测点吻合，但运算量较大。

由于难以确定样条插值拟合与多项式拟合的最低成本，将在两种尝试后进行选择。

3. 启发式搜索

实验目的是对给定目标函数求得最优解，通过枚举法计算由于计算量太大难以找到解，启发式算法可以在合理的时间内找到不错的解，我们决定使用遗传算法进行搜索。

3.1 遗传算法介绍

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。遗传算法的实现可分为以下几个步骤：首先随机建立初始状态即个体基因，根据实际问题求解函数来设定适应度函数。其次对每个个体适应度函数的求取，根据适应度值的大小进行轮盘概率求取，个体间进行交叉和变异，繁殖出具有适应度更好的下一代，再次进行选择直至代数达到上限。 遗传算法的原理是模式定理与积木块假设。

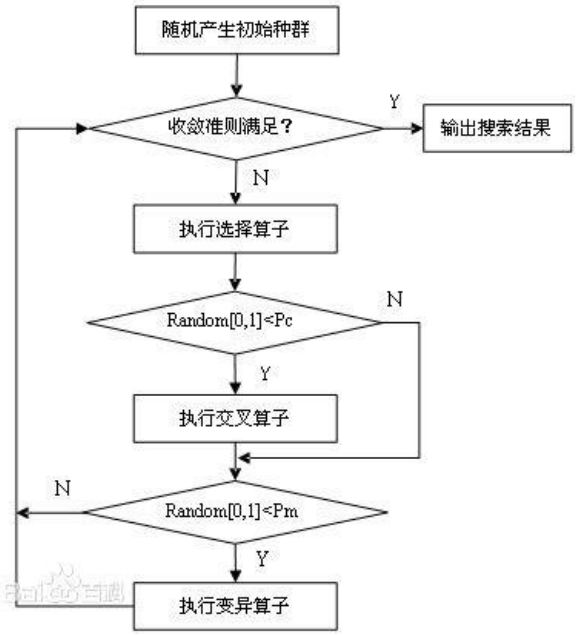

图 3 遗传算法流程图

3.2 遗传算法及程序设计思路

（1） 初始化：设立初始种群，具有 M 个个体，每个个体的基因长度为 51，代表了一种取点方案，每个基因点是 0 或 1 连个状态代表选取该位置点或不选取该点。

（2） 个体评价：由选定的拟合方式计算每一个个体代表的取点方案的定标成本，包括误差成本和取点成本的两项内容。对于适应度的选择评判，我们选择用倒数来刻画，对种群中的每个个体拟合得到的综合成本取倒数，成本小的适应度高，应该被保留，大的适应度低，应当被淘汰。这样就求出了每个个体所对应的适应度，并且满足误差小的适应度高的条件。但在实际计算中发现 1/x 随 x 增大变化较慢，最后达到最优结果前收敛较慢，所以参考资料改用 1/（x-a）函数刻画。a 由估计 x 下限决定。

（3） 交叉算子和变异算子：对于为了配对时的方便，在适应度计算完毕后，增加了一个赌轮选择出过渡种群：对于适应度矩阵，从第二项开始，进行累加，使得每一项和前一项构成一个区间，形成一个赌轮上的区域。再用随机数生成 M 个随机数矩阵，对于这 M 个数每个都和每一个赌轮区间进行比较，如果落在区间里，就记录对应的父代的位置。然后令父代中取点方法进行交换，产生子代，完成交叉。在产生子代过程中，为了引入新的取点方式，引入变异算子，产生随机数矩阵，与变异概率大小进行比较，如果进行变异，则选择将某随机点在 1 和 0 间变换，完成变异。

（4） 终止条件判断：达到终止代数或最优解时停止运算，否则回到步骤 2 进行个体评价。

3.3 遗传算法的具体实现

本次实验利用了 matlab 进行编程实现，利用了其矩阵运算的优点，使用了模块化的编程思想，拆解出多个函数。遗传算法各部分的实现如下：

（1） 初始化：由于取点数太多则成本高，太少则误差成本高，初始化时决定随机取出 7 个点作为初始种群，生成指定个数 N 个 1*51 元素为 0，1 的矩阵.初始化直接在遗传算法函数 fitness（）中。

（2） 个体评价：对于三次多项式的拟合利用自带函数 polyfit（）拟合出曲线，使用成本函数计算，返回成本。对于三次样条插值拟合利用 interp（）函数进行拟合，使用成本函数计算，返回成本。

（3） 交叉运算和变异运算：为对应于各基因，利用一个 1*51 的矩阵用 0 和 1 表示取点方案，利用个体评价进行筛选出父代，根据概率交换部分基因产生后代，可以采用整体交换或均匀交换。变异即是利用 1 减去值即改变取点方案，对应位置交叉产生子代。为避免陷入局部最优解，在最优解多代不变时积极进行干预改变，如改变交叉和变异概率，引入新个体。每十代输出运算结果进行观察。

（4） 终止条件：当达到指定代数时停止运算，输出此时得到的最优解及成本。

函数介绍：

（1） fitness（num,gen,crossRate,mutaRate） 为主函数,初始化种群，num 为个体个数，gen 为进化代数，crossRate 为交叉概率，mutaRate 为变异概率，返回最优解。

（2） spline3（program） program 为一个个体的取点方法，利用三次样条插值方法计算并返回成本。

（3） polyfit3（program） program 为一个个体的取点方法，利用三次多项式拟合方法计算并返回成本。

（4） n_T-b（t） t 是十进制的取点方法，将其转化为二进制表示。

（5） b_T_n（t） t 是二进制的取点方法，将其转化为十进制表示。

3.4 遗传算法的具体问题

优秀的遗传算法应该尽快收敛到最优解,对遗传算法收敛性产生影响的主要因素有种群规模，适应度函数，交叉与变异概率。

（1）种群规模：种群规模太小难以提供足够的取样点信息，造成算法性能低下，规模太大使计算的复杂度大大提高，收敛缓慢，造成时间的消耗。

（2）适应度函数：遗传算法中适应度函数的选择决定了对亲代的选择，显然令接近最优解的个体具有更高的适应度将加快进化的速度。当计算得成本为 c 时，成本越高存活率越低，我们令 1/c 为适应度函数可以满足要求，但在实际应用中函数收敛速度较慢，如果可以估计得到成本最低为 a,令适应度函数为 1/（c-a）将加大区分度，加快进化进程。同理适应度函数可以取为指数函数等，但运算代价较高。同时如果能够保留父代最优解将加快收敛进程。

（3）函数参数的影响：函数参数对进化过程产生影响，当个体数 num 较大时运算较慢，进化也由于较优解比例太低难以进化，交叉概率和变异概率太小难以实现基因的进化，太大又难以保留最优解。Schaffer 的建议参数是 num 为 20~100，gen 为 100~500，crossRate 为 0.4~0.9，mutaRate 为 0.001~0.01.。经过查阅资料和测试后，我们选择 30 个体，交叉概率 0.8，变异概率 0.2 进行进化

（4）跳出局部最优解：在运算中我们令其每十代输出结果，分析发现易陷入局部最优解，即最优解多代不变，而局部最优解在种群中占的比例太大难以进化。我们可以采取人工干预的方法进行优化。用 best 记录最优解，当多代最优解相同时，我们增加交叉与变异的概率，使个体进化的概率增加。同时引入新的个体，保持种群的多样性。

4 算法结果及分析

借助 matlab 编程初步完成了利用遗传算法进行最优选点，得到结果如下：

（1） 三次样条插值拟合:，多次运行得到最优解：

方案：2 10 20 26 33 43 49

成本：95.7146

（2） 三次多项式拟合：多次运行得到最优解：

方案：3 15 27 38 49

成本：114.9233

从多次结果比较可知样条插值拟合的效果好于多项式拟合。从拟合方法的角度分析，由所给样品图线上可知数据图像大致分为三段，每段斜率不同，多项式表达是为了解得特定多项式表达式，而图线方程不一定契合多项式表达，多项式拟合存在一定误差。所以三次样条插值较优的结果是可以接受的。

本次结果是在不确定取点个数的情况下进行个体进化，实际在多次运行程序后可以发现最优取点个数是 7 个点，可以将代码优化为确定取点个数的进化，可以发现可以大大加快进化速度。

5. 问题思考

在遗传算法实现中遇到的问题主要是局部最优解的出现。局部最优解的出现是由于在进化过程中局部最优基因在种群中所占比例太高，交叉失去意义，而局部最优解附近的解适应度相似，变异也失去了作用，对此我们必须采取外界干预的手段来使种群进化，否则将付出巨大的时间代价来跳出局部最优解。我们认为有以下手段可以使用：

(1) 初始化种群时基因应随机取得，基因的多样性可以降低陷入局部最优解的概率。种群数量应适当，太小不足以产生最优解，太大由于新基因比例太小难以起到作用，进化缓慢，而且变异的价值降低。

(2) 可以适当更改适应度函数，使其在最优解附近仍有较大区分度，这样在进化后期任然可以有较快的收敛速度。为增加最优解的生存度，可以将生存度函数由 1/(x-a)更改为 1/（1-x）^n，经测试可以提高收敛速度，但 n 较大时运算复杂而缓慢。。同时在难以找到较优函数时可以使用适应度表格，将每个成本对应一个适应度，人为控制个体的生存概率，从而增加较优解的生存概率从而跳出局部最优解，但代价较高，在成本复杂时不宜使用。

(3) 执行交叉算子时可以不是分段交叉，而是随机均匀交叉。陷入局部最优解时分段交叉对如果交换部分相似，则相似解意义不大，可能出现大部分个体所交换基因部分相似。均匀交叉可以使基因交换位置均匀，且操作简单。

(4) 可以在陷入局部最优解时，将变异概率与交叉概率尽量提高以产生新的基因，或者在局部最优解中可以随机引入新的个体从而产生新的基因。当局部最优解受到新基因的较大扰动时有更大的几率发生进化，从而跳出局部最优。

6 参考文献

[1]统计推断：第 nn 组（组长姓名）课程报告设计

[2]“统计推断”课程设计的要求
[3]王志新 matlab 程序设计及其数学建模应用 科学出版社
[4]维基百科 遗传算法

附录

遗传算法代码

1. 主函数

fitnee.m

（此函数使用样条插值拟合，换为多项式拟合时只需将调用函数 spline3()换为 polyfit3().）

```
function s = fitness( num,gen,crossRate,mutaRate )

%遗传算法寻找最优解

%num 种群个体数 gen 遗传代数 rate 变异几率

tic;
pop=zeros(num,51);
pop_new=zeros(num,51);
liveTable=zeros(1,num);

x = randi(51,num,6);
for i=1:num
  for j=1:6
     pop(i,x(i,j))=1;
  end
end                          %产生初代种群


time=0;
best=pop(1,:);
last=best;
best_n=b_T_n(best);

mm = mutaRate;
cc = crossRate;
while time<gen
    mutaRate = mm;
    crossRate = cc;
    time=time+1;
    totalFit=0;
```

```
    for u=1:num
        program=b_T_n(pop(u,:));
        if(length(program)<=3 || length(program)>=50)
              liveTable(u)=0;                    %防止极端取点方式
        else
           t=spline3(program);
           liveTable(u)=1000/(t-50);

        end
        totalFit=totalFit+liveTable(u);
    end

    liveTable=liveTable/totalFit;          %计算存活度储存在 live
    [~,s]=max(liveTable);
    best=pop(s,:);                                    %最优

    if last == best
        mutaRate=0.4;
        crossRate=0.9;                %陷入局部最优时增加变异
    end



    if spline3(b_T_n(best))<spline3((b_T_n(last)))   %储存最
优解，减少退化
       last=best;
    end



    fitTable=cumsum(liveTable,2);
    x=rand(1,num);
    for u=1:num
        pos=find(fitTable>x(u));
        pop_new(u,:)=pop(pos(1),:);
    end
    pop=pop_new;                                  %利用生存概率赌轮盘法
进行随机选择亲代
```

```
    for u=1:2:num
        t = rand;
        if t<crossRate
            t=randi([0,1],[1,51]);
            for j=1:51
                if t(j) == 1
                  pop_new(u,j) = pop(u+1,j);
                  pop_new(u+1,j) = pop (u,j);
                end
            end
        end
    end
    pop=pop_new;                %相邻个体进行配对互换，用 pop 存储
    pop(num,:)=last;

    for u=1:num-1
        t=rand;
        if t<mutaRate
          rany=randi(51);
          pop(u,rany)=1-pop(u,rany);
        end
    end                 %加入变异




    if mod(time,10)==0
    best_n=b_T_n(last);
    fprintf('generation: %d\n',time);
    fprintf('best of the generation: %d\n',best_n);
    fprintf('value of the best:%d\n',spline3(best_n));


    end                                 %每十代输出结果

end
fprintf('best of the all the generations:\n');
```

```
disp(best_n);
fprintf('value of the best\n');
disp(spline3(best_n));

toc
end
```

2. 三次插值线条拟合

Spline3.m

```
function cost = spline3( program )
%计算三次插值线条拟合成本

%  program 取点方法
M =zeros(800,51);
M = csvread('20150915dataform.csv');  %读入数据


x = zeros(800,51);
y = zeros(800,51);
p = zeros(800,51);                    %储存三次多项式拟合的结果

expect = zeros(800,51);               %储存拟合函数中预期的值

for j=1:400
    y(j,:)=M(2*j,:);
    x(j,:)=M(2*j-1,:);
end
  X=x(:,program);
  Y=y(:,program);

for j=1:400                                          %三次多项式拟
合
    p(j,:) = interp1(X(j,:),Y(j,:),x(j,:),'spline');
end

err=abs(p-y);                                        %计算成本

a1=length(find((err<=0.6 & err>0.4)));
a2=length(find((err<=0.8 & err>0.6)));
a3=length(find((err<=1 & err>0.8)));
a4=length(find((err<=2 & err>1)));
a5=length(find((err<=3 & err>2)));
a6=length(find((err<=5 & err>3)));
```

```
a7= length(find((err>5)));
C=a1*0.1+a2*0.7+a3*0.9+a4*1.5+a5*6+a6*12+a7*25;
cost=sum(sum(C,2))/400+12*length(program);

end
```

3. 三次多项式拟合

Polyfit3.m

```
function cost = polyfit3 (program)
%   计算三次多项式拟合的成本

%   program 取点方案 cost 成本
M =zeros(800,51);
M = csvread('20150915dataform.csv');  %读入数据


x = zeros(800,51);
y = zeros(800,51);
p = zeros(800,4);                    %储存三次多项式拟合的结果

expect = zeros(800,51);              %储存拟合函数中预期的值
for j=1:400
    y(j,:)=M(2*j,:);
    x(j,:)=M(2*j-1,:);
end
  X=x(:,program);
  Y=y(:,program);
for j=1:400                                                  %三次多项式拟
合
    p(j,:) = polyfit(X(j,:),Y(j,:),3);
  for k=1:51

expect(j,k)=p(j,1)*(x(j,k))^3+p(j,2)*(x(j,k))^2+p(j,3)*x(
j,k)+p(j,4);
  end
end
err=abs(expect-y);                                           %计算
```

成本

```
a1=length(find((err<=0.6 & err>0.4)));
a2=length(find((err<=0.8 & err>0.6)));
a3=length(find((err<=1 & err>0.8)));
a4=length(find((err<=2 & err>1)));
a5=length(find((err<=3 & err>2)));
a6=length(find((err<=5 & err>3)));
a7= length(find((err>5)));
C=a1*0.1+a2*0.7+a3*0.9+a4*1.5+a5*6+a6*12+a7*25;
cost=sum(sum(C,2))/400+12*length(program);

end
```

4. 取点表示形式的转化

b_T_n.m

```
function q = b_T_n( t )
%将取点方法由二进制转为十进制

%   t为二进制，51元矩阵
j=0;
q=zeros(1,1);
for i=1:51
  if t(i)==1
      j=j+1;
      q(j)=i;
  end
end
end
```

n_T_b.m

```
function q = n_T_b( t )
%将取点方法由十进制变为二进制

%   t为十进制矩阵
q=zeros(1,51);
q(t)=1;

end
```