单位代码	10476
学　　号	1401183029
分 类 号	O212.6

河南師範大學

硕士学位论文

正交表构造及其在数值计算的应用的 Matlab 实现

学科、专业：概率论与数理统计
研究方向：试验设计
申请学位类别：理学硕士
申　请　人：鹿姗姗
指导教师：庞善起　教授

二〇一七年三月

The Construction and Application in Numerical Calculation of Orthogonal Arrays by Matlab

A Dissertation Submitted to
the Graduate School of Henan Normal University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science

By

Lu Shanshan

Supervisor: Pang Shanqi

2017. 3

摘 要

本文首先根据已有的正交表的构造方法和原理并利用Matlab软件实现正交表的构造. 比如利用哈达玛直积和特征函数生成$N = 2^s$型正交表; 利用正交拉丁方生成$L_{t^2}(t^m)$型正交表; 再利用差集矩阵和向量内积分别生成$L_{\lambda p^2}(p^{\lambda p+1})$型和$L_{t^u}(t^m)$型正交表; 最后利用并列法、递归构造分别生成混合水平正交表和高强度正交表.

然后把非线性方程组和超定方程组的求解问题转化为函数极值的求解问题; 再根据正交表区间收缩法求极值的思想, 在一定的改进下, 安排正交试验进而求得非线性方程组和超定方程组的解或者近似解. 区间收缩法的优势在于不用通过求导就可以求得任何函数的极值点. 这种方法解方程组, 用时少, 精确度高. 特别地, 矩阵是$Matlab$语言的基本的运算单元, 所以使用$Matlab$运算矩阵有较大的优势. 通常用C语言难以实现或者实现过程很复杂的, 在$Matlab$中都能很容易的经过矩阵实现正交表的构造.

本文主要研究了如何利用计算机快速生成正交表,及其在数值计算方面的应用.. 全文共分为四章:

第一章介绍了全文的研究背景、相关概念和已有研究成果.

第二章根据正交表的构造方法和原理, 利用$Matlab$语言生成对称正交表、混合水平正交表和高强度正交表.

第三章研究了如何在$Matlab$中利用区间收缩法将一类方程组的求解转化为函数求极值.

第四章对本篇论文作出总结, 并提出意见和建议.

关键词: 正交表, 区间收缩法, 非线性方程组, 超定方程组, $Matlab$

Abstract

This paper constructs the orthogonal arrays(OAs) based on the method and principle by Matlab software. For example, the model of $N = 2^s$ of OAs is constructed by Hadamard matrix and eigenvalue function; the model of $L_{t^2}(t^m)$ of OAs is constructed by design of orthogonal latin squares. And then we construct the model of $L_{\lambda p^2}(p^{\lambda p+1})$ and $L_{t^u}(t^m)$ of OAs by difference matrix and inner driving drum product of vector. Finally, mixed-level OAs and OAs of high strength are constructed by parallel principle and recursive construction.

The problem of solving nonlinear equations and over-determined equations is transformed into the problem of selecting the function extreme value point. And then, according to the interval shrinkage of OAs, we do the orthogonal experiment to select the nonlinear equations under a certain improvement. Extreme value point of any function can be obtained by derivative extremum. This method takes less time to get high precision. There are notable advantages for using matlab language program on the matrix operations since its basic computing unit is the matrix. Furthermore, compared with the C language, some operations are implemented much easier by Matlab.

This paper mainly studies the construction and application in numerical calculation of OAs by Matlab. The paper consists of four chapters and its structure is as follows.

Chapter 1 lists the construction method and principle to generate the model of symmetric, mixed-level and high strength of OAs by matlab software.

Chapter 2 lists some basic constructions of mixed orthogonal arrays.

Chapter 3 studies how to use interval shrinkage to transfer solution of equations to seeking extremum of function.

Chapter 4 summarizes the main content of this paper and comes up with some suggestions.

Keywords: orthogonal arrays, interval shrinkage, nonlinear equation systems, overdetermined systems, Matlab

目 录

第一章　绪 论

§1.1　学科历史及研究现状

1978年$C.R.Rao$引入了正交表, 它不仅在试验设计中有广泛的应用, 而且在卫生、统计、医药、农业、制造业等方面都有非常重要的应用, 还可以被应用于编码学、密码学、计算机科学中. 尤其是混合水平正交表, 它们的优势就是有很大的灵活性, 容许试验因素具备不同水平数, 另外, 高强度正交表可以研究较多因素之间的交互作用, 得到了越来越多数学家和统计学家的关注和研究.

1999年, 杨子胥编著并出版了有关正交表的著作《正交表的构造》, 如文献[17]; 2004年庞善起编写了《正交表的构造方法及其应用》, 如文献[12]; 2012杜蛟等发表了《强度m的对称正交表的递归构造》, 如文献[5]; 以及近几年庞善起教授领导自己的学生专门研究正交表的各种构造方法, 从而生成各种类型的正交表, 例如文献[13, 14]. 但是至今仍然缺少有关系统的介绍如何利用计算机软件快速、高效的得到需要的正交表的文章, 因此本文利用$Matlab$语言生成多种类型的正交表具有重要意义.

在实际应用中, 很多数学问题都转化为方程组的求解问题, 如天气预报、石油地质勘探、计算力学、计算生物化学、优化控制和轨道设计等, 如文献[26]. 从实际应用这一方面考虑, 寻找高效可靠的算法是有意义的. 为此人们作了大量的研究. 但是非线性方程求解仍然是困扰人们的一个难题, 尤其是实际应用问题, 始终缺乏高效可靠的算法. 以牛顿迭代法为代表的求解方法依赖于初始点的选择, 不合适的初始点很容易导致算法收敛失败; 然而如何选择一个好的初始点, 是一件有一定难度的事情, 而本文算法完全不需要初始点.

§1.2　预备知识

下面介绍本文中要用到的定义:

定义1.2.1. *([12])* 一个第j列的元素是$0, 1, \cdots, q_j-1$的$\omega \times n$矩阵A, 称为一个强度为2的正交表, 如果满足以下条件:

*(1)*每一列中每个元素出现的次数相同;

*(2)*在任两列a_i, a_j中的每一个数对*(0,0)*, $\cdots$, *(0,* q_j-1*)*, *(1,0)*, $\cdots$, *(1,* q_j-1*)*, $\cdots$, *(*q_j-1*,* *0)*, $\cdots$, *(*q_j-1*,* q_j-1*)* 出现的次数相同.

如果$q_1, \cdots, q_s$中有一些相同, 用$L_n(q_1^{t_1}\cdots q_m^{t_m})$ 表示, 其中$t_1+t_2+\cdots+t_m=s$, 可以得到以下结论:

*(1)*如果$m=1$, 即所有的列具有相同的水平, 称$L_n(q_1^{t_1}\cdots q_m^{t_m})$ 为一个对称正交表.

*(2)*如果$m\geq 2$, 称$L_n(q_1^{t_1}\cdots q_m^{t_m})$ 为一个混合水平正交表, 或者非对称正交表.

*(3)*如果A的任何$\omega\times d$的子矩阵包含所有可能的$\omega\times d$行向量, 且这些向量出现的次数相同,称为强度为d的正交表.

*(4)*强度$d\geq 3$的正交表称为高强度正交表, 对任意的$0\leq k\leq d$, 显然强度d 的正交表都是强度k的正交表.

定义1.2.2. *([17])* 以± 1为元素, 并且在任二列正交的矩阵, 称为哈达马矩阵, 简称哈阵.

定义1.2.3. *([17])* 对任意一个哈阵, 总可以用对行乘-1的方法, 把它的第一列变成全1列, 我们称这样的哈阵为标准的哈阵, 并把这个过程叫做标准化.

定义1.2.4. *([17])* 设A, B分别为$n\times m$, $s\times t$矩阵, 将A的每个元素都乘以矩阵B而得到的$ns\times mt$矩阵, 用$A\otimes B$表示, 叫做矩阵A与B 的直积.

引理1.2.5. *([17])* 标准哈阵去掉全*1* 列后, 便是一个二水平的正交表; 反之, 二水平正交表添上全1列后, 便是一个标准的哈阵.

定义1.2.6. *([19])* 对设计$H=(a_{ij})_{n\times m}$ 可以定义H的因子x_j的水平k的示性集合为

$$H_{jk}=\{i: a_{ij}=k, i=1,\cdots,n, j=1,\cdots,m\}$$

H_{jk}中所含元素个数记为$r_{jk}=|H_{jk}|$, 而H_{jk}是设计H的第j列的水平k 的行下标因子. x_j水平k的水平均值$\mu_{jk}=\frac{1}{r_{jk}}\sum_{i\in H_{jk}} f(x_1^i, x_2^i, \cdots, x_m^i)$.

第二章　正交表的构造方法及$Matlab$实现

本章节主要引用矩阵直积法、特征函数法、正交拉丁法、差集矩阵法、向量内积法、并列法、递归构造法等, 在$Matlab$软件中生成对称正交表、混合水平正交表和高强度正交表.

§2.1　$N=2^s$型正交表

哈阵标准化后, 去掉全1列就是2水平的正交表, 因此2水平正交表可以转化为构造哈阵, 在本章节中引用两种方法构造哈阵: (1)利用矩阵直积构造哈阵; (2)利用特征函数构造哈阵, 具体情况如下:

第一种方法: 哈达玛直积法

引理2.1.1. *([17])* 哈阵的直积仍为哈阵, 即A和B是哈阵, 那么$A\otimes B$也是哈阵.

由引理2.1.1可知, 可以从二阶哈阵出发, 利用矩阵直积法, 一步一步地生成高阶的哈阵, 再去掉全1 列, 也就是第一列, 即可得到$L_N(2^{N-1})$型正交表. 具体例子如下:

例2.1.2. 以正交表$L_8(2^7)$为例,从最简单的二阶哈阵出发, 利用直积, 逐步地构造出8阶的哈阵, 去掉全1列, 即可得到正交表$L_8(2^7)$. 结果如表$A-1$, $Matlab$程序与解释如下:

```
clear;
N = 8 %初始条件
H = [1 1;1 -1]; %最简单的二阶哈达玛矩阵
h = [1 1;1 -1]
i = 2;
while(i <= N/2)
 I = ones(2);
 H = kron(H, I); %直积
 for j = 1 : i
```

$for\ k = 1 : i$

$n = (1 + (k-1) * 2) : k * 2;$

$m = (1 + (j-1) * 2) : j * 2;$

$H(n, m) = H(n, m). * h;$

end

end

$i = i * 2;$

end

$[s\ t] = size(H);$

$L = H(:, 2 : t);$

$L(L == 1) = 0;\ L(L == -1) = 1;$

L % 得到常见的$L_8(2^7)$

在$Matlab$程序中只需改变初始条件N的值, 就可以生成$L_N(2^{N-1})$型正交表, 例如当$N = 4$时, 生成$L_4(2^3)$; 当$N = 16$时, 生成$L_{16}(2^{15})$; 当$N = 64$时, 生成$L_{64}(2^{63})$.

第二种方法: 特征函数法

定义2.1.3. *([17])* 设α为有限域$GF(p^n) = \{0, \alpha^0, \alpha^1, \alpha^2, \cdots, \alpha^{p^n-2}\}$的一个元根, 在$GF(p^n)$上定义一个函数$\psi(x)$, 它的取值规定为:

$$\psi(0) = 0, \psi(\alpha^s) = \begin{cases} 1, & \text{当}s\text{为偶数;} \\ -1, & \text{当}s\text{为奇数,} \end{cases}$$

我们称$\psi(x)$为定义在$GF(p^n)$上的一个特征函数.

引理2.1.4. *([17])* 对p阶有限域R_p来说, 若适当排列元素的次序, 例如排为$R_p = \{p = 0, p-1, p-2, \cdots, 2, 1\}$, 则所得的$R$阵是一个第一行为$\psi(0), \psi(p-1), \psi(p-2), \cdots, \psi(2), \psi(1)$

的p阶循环矩阵,

$$R=\begin{pmatrix} 0 & \psi(p-1)\psi(p-2)\cdots\psi(2)\psi(1) \\ \psi(1) & 0 \quad \psi(p-1)\cdots\psi(3)\psi(2) \\ \psi(2) & \psi(1) \quad 0 \quad \cdots\psi(4)\psi(3) \\ \cdots & \cdots \quad \cdots \quad \cdots\cdots\cdots \\ \psi(p-1)\psi(p-2)\psi(p-3)\cdots\psi(1) & 0 \end{pmatrix}$$

以R阵为基础, 分以下两种情形来构造哈阵, 即$p^n\equiv 1(mod4)$和$p^n\equiv -1(mod4)$, 如文献[17]所示, 具体情况如下:

第一种情况: 当$p^n\equiv 1(mod4)$, 且p是奇素数时, 构造$2(p^n+1)$阶哈阵, 标准化后去掉全1列, 即可得到正交表$L_{2(p^n+1)}(2^{2p^n+1})$, 基本的方法和步骤如下:

第一步: 给出$GF(p^n)=\{a_1,a_2,\cdots,a_{p^n}\}$;

第二步: 找到$GF(p^n)$的一个元根α, 并把域中每个非零元素都表示为元根的方幂的形式, 如$GF(p^n)=\{0,\alpha^0,\alpha^1,\alpha^2,\cdots,\alpha^{p^n-2}\}$, 并求出$\psi(0),\psi(\alpha^0),\psi(\alpha^1),\psi(\alpha^2),\cdots,\psi(\alpha^{p^n-2})$的值, 作为$R$阵的第一行, 由引理2.1.4求得$R$阵;

第三步: 构造K阵, $K=\begin{pmatrix} 0 & e' \\ e & R \end{pmatrix}$, 其中$e=(1,1,\cdots,1)^T$, K是p^n+1阶的矩阵;

第四步: 构造$2(p^n+1)$阶的哈阵$H_{2(p^n+1)}$, $H_{2(p^n+1)}=\begin{pmatrix} K+E & K-E \\ K-E & -K-E \end{pmatrix}$;

第五步: 将$H_{2(p^n+1)}$标准化, 去掉全1列, 即可得到正交表$L_{2(p^n+1)}(2^{2p^n+1})$.

例2.1.5. 以$L_{12}(2^{11})$为例, 则$2(p^n+1)=12$, 此时$p=5$是奇素数, 且$5\equiv 1(mod4)$, 构造12阶哈阵, 去掉全1列, 即可生成正交表$L_{12}(2^{11})$, 结果如表$A-2$, $Matlab$程序与解释如下:

```
clear;
n=1,p=5,m=p^n %初始条件
g=2 %p^n=5阶域的元根
```

%以下生成特征函数

$X = mod(0:(m-2),2)$ %$p^n = 5$阶域的非零元素的指数

$for\ i = 1:(m-1)$

$if\ (X(i) == 0)$

$X(i) = 1;$

$else$

$X(i) = -1;$

end

end

%以下生成循环矩阵R

$G = [0\ X]$ %R的第一行

$R = zeros(m);$

$forj = 1:m$

$R(j,:) = circshift(G,[0,j-1]);$

end

R

$e = ones(m,1)$ %m行1列元素全为1的矩阵

$K = [0\ e';e\ R]$

$E = eye(m+1,m+1)$ %单位矩阵

$H = [K+E, K-E; K-E, -K-E]$

$H(find(H(:,1) == -1),:) = -1 * H(find(H(:,1) == -1),:)$ %标准化

$[s\ t]$=size(H); %哈阵的行数和列数

$L = H(:,2:t)$; %去掉第一列

$L(L==1)=0; L(L==-1)=1$; %L中所有的1, -1的元素替换成0, 1,

L %得到常见的正交表$L_{12}(2^{11})$

在此$Matlab$程序中只需改变初始条件p, n和元根g的值, 即可得到形如$L_{2(p^n+1)}(2^{2p^n+1})$的正交表, 如$n=1$, $p=13$, 此时$g=2$, 生成$L_{28}(2^{27})$; $n=1$, $p=17$, 此时$g=3$, 生成$L_{36}(2^{35})$; $n=1$, $p=73$, 此时$g=5$, 生成$L_{148}(2^{147})$.

第二种情况: 当$p^n \equiv -1(mod4)$, 且p是奇素数时, 构造p^n+1阶哈阵, 标准化后去掉全1列, 从而得到正交表$L_{p^n+1}(2^{p^n})$, 如文献[17]所示, 基本的方法和步骤如下:

1. 给出$GF(p^n)=\{a_1, a_2, \cdots, a_{p^n}\}$;
2. 找出$GF(p^n)$的一个元根α, 并把域中每个非零元素都表为元根的方幂, 形如$GF(p^n)=\{0, \alpha^0, \alpha^1, \alpha^2, \cdots, \alpha^{p^n-2}\}$, 并求出$\psi(0), \psi(\alpha^0), \psi(\alpha^1), \psi(\alpha^2), \cdots, \psi(\alpha^{p^n-2})$的值, 作为$R$阵的第一行, 由引理2.1.4求得$R$ 阵;
3. 作出K阵, $K=\begin{pmatrix} 0 & e' \\ -e & R \end{pmatrix}$, 其中$e=(1,1,\cdots,1)^T$, K阵是p^n+1 阶的矩阵;
4. 作出p^n+1阶的哈阵, $H_{p^n+1}=K+E$;
5. 将H_{p^n+1}的第一行乘以-1, 然后将第3行与第p^n+1行交换, 第4行与第p^n行交换, $\cdots$, 便可得到一个对称的哈阵. 再将其标准化, 再去掉全1列, 即可得到$L_{p^n+1}(2^{p^n})$ 型正交表.

例2.1.6. 以$L_{20}(2^{19})$为例, 则$p^n+1=20$, 此时$p=19$ 是奇素数, 且$19 \equiv -1(mod4)$, 构造20阶哈阵, 去掉全1列, 即可生成正交表$L_{20}(2^{19})$, 结果如表$A-3$, $Matlab$程序与解释如下:

```
clear;
n=1,p=19 %初始条件
m=p^n
g=2 %p^n=19阶域的元根
%以下生成特征函数
```

$X = mod(0:(m-2),2)$ %$p^n = 19$阶域的非零元素的指数

$for\ i = 1:(m-1)$

$if(X(i) == 0)$

$X(i) = 1;$

$else$

$X(i) = -1;$

end

end

%以下生成循环矩阵R

$G = [0\ X]$ %R的第一行

$R = zeros(m);$

$for j = 1:m$

$R(j,:) = circshift(G,[0,j-1]);$

end

R

$e = ones(m,1)$ %m行1列元素全为1的矩阵

$K = [0\ e';-e\ R]$

$E = eye(m+1,m+1)$ %单位矩阵

$H = K + E$

$H(1,:) = -1 * H(1,:)$ %第一行$*-1$

$H1 = H(1:2,:)$

$H2 = flipud(H(3:(m+1),:))$ %行

$H = [H1;H2]$

$H(find(H(:,1)==-1),:)=-1*H(find(H(:,1)==-1),:)$ %标准化

$[s\ t]$=size(H); %哈阵的行数和列数

$L=H(:,2:t)$; %删除4阶哈阵的第一列, 剩余的就是$L_{20}(2^{19})$

$L(L==1)=0$; $L(L==-1)=1$; %L中所有的1, -1的元素替换成0, 1,

L %得到常见的正交表

在$Matlab$程序中只需改变初始条件p, n和元根g的值, 我们就生成形如$L_{p^n+1}(2^{p^n})$ 的正交表, 如$n=1$, $p=7$, 此时$g=3$, 生成$L_8(2^7)$; $n=1$, $p=11$, 此时$g=2$, 生成$L_{12}(2^{11})$; $n=1$, $p=83$, 此时$g=2$, 生成$L_{84}(2^{83})$.

§2.2 $L_{t^2}(t^m)$型正交表

$L_{t^2}(t^m)$型正交表的构造可以转化为构造正交拉丁方完全组, 再将其中的每一个正交拉丁方都按行展开, 添上基本列$(\underbrace{1,\cdots,1}_{t},\underbrace{2,\cdots,2}_{t},\cdots,\underbrace{t,\cdots,t}_{t})$和$(\underbrace{1,2,\cdots,t}_{t},\underbrace{1,2,\cdots,t}_{t},\cdots,\underbrace{1,2,\cdots,t}_{t})$, 即可得到$L_{t^2}(t^m)$型正交表. 本文中采用以下三个方法构造正交拉丁方完全组:

引理2.2.1. *([17])* 设n为任意一个素数或素数幂, $GF(n)$为n阶有限域, 而$a_1,a_2,\cdots,a_n$与$b_1,b_2,\cdots,b_n$为域中全体元素的任意两个排列; $c_1,c_2,\cdots,c_{n-1}$为域中的全体非零元素, 则所有方阵

$$A_i=\begin{pmatrix} a_1+b_1c_i & a_2+b_1c_i & \cdots & a_n+b_1c_i \\ a_1+b_2c_i & a_2+b_2c_i & \cdots & a_n+b_2c_i \\ \cdots & \cdots & \cdots & \cdots \\ a_1+b_nc_i & a_2+b_nc_i & \cdots & a_n+b_nc_i \end{pmatrix},$$

其中$i=1,2,\cdots,n-1$, 构成n阶正交拉丁方完全组.

第一种方法: 由引理2.2.1得到的正交拉丁方A_i按行展开, 添上基本列$(\underbrace{1,\cdots,1}_{t},\underbrace{2,\cdots,2}_{t},\cdots,\underbrace{t,\cdots,t}_{t})$和$(\underbrace{1,2,\cdots,t}_{t},\underbrace{1,2,\cdots,t}_{t},\cdots,\underbrace{1,2,\cdots,t}_{t})$, 即可得到$L_{t^2}(t^m)$型正交表, 具体例子如下:

例2.2.2. 以$L_9(3^4)$为例, 此时$t=3$, 构造3阶正交拉丁方完全组, 添加基本列从而得到正交表$L_9(3^4)$, 结果如表$A-4$, $Matlab$ 程序与解释如下:

$clear;$

$t=3;$%初始条件

$m=t-1; n=t^2$

$for\ j=0:m$

$for\ k=0:m$

$a(k+1,j+1)=j;$

$b(k+1,j+1)=k;$

end

end

$a;\ b;$

$L1=kron((0:m)', ones(t,1));$ % 基本列第一列

$L2=kron(ones(t,1),(0:m)');$ % 基本列第二列

$L=[L1, L2, ones(n,m)];$

$i=1;$

$while(i<=m)$

$A=mod(a+i*b,t);$ % 正交拉丁方完全组

$H=reshate(A',n,1);$ % 正交拉丁方完全组按行展开

$L(:,2+i)=L(:,2+i,:).*H;$ %.*表示对应位置的元素相乘

$i=i+1;$

end

L

在$Matlab$程序中只需改变初始条件t的值, 我们就生成$L_{t^2}(t^m)$ 型正交表.

引理2.2.3. *([17])* 设α为n阶有限域$GF(n)$的一个元根，则所有方阵

$$B_{i+1}=\begin{pmatrix} 0 & 1 & \alpha & \cdots \alpha^{n-2} \\ \alpha^{i} & 1+\alpha^{i} & \alpha+\alpha^{i} & \cdots \alpha^{n-2}+\alpha^{i} \\ \alpha^{i+1} & 1+\alpha^{i+1} & \alpha+\alpha^{i+1} & \cdots \alpha^{n-2}+\alpha^{i+1} \\ \cdots & \cdots & \cdots & \cdots\cdots \\ \alpha^{i+n-2} & 1+\alpha^{i+n-2} & \alpha+\alpha^{i+n-2} & \cdots \alpha^{n-2}+\alpha^{i+n-2} \end{pmatrix},$$

其中$i=0,1,2,\cdots,n-2$, 构成n阶正交拉丁方完全组.

第二种方法: 由引理2.2.3得到的正交拉丁方A_i按行展开, 添上基本列$(\underbrace{1,\cdots,1}_{t},\underbrace{2,\cdots,2}_{t},\cdots,\underbrace{t,\cdots,t}_{t})$和$(\underbrace{1,2,\cdots,t}_{t},\underbrace{1,2,\cdots,t}_{t},\cdots,\underbrace{1,2,\cdots,t}_{t})$, 即可得到$L_{t^2}(t^m)$型正交表, 具体例子如下:

例2.2.4. 以$L_{25}(5^6)$为例, 此时$t=5$,构造5阶正交拉丁方完全组, 从而生成正交表$L_{25}(5^6)$, 结果如表$A-5$, $Matlab$程序与解释如下:

```
clear;
t = 5; %初始条件
m = t − 1; n = t^2
g = 2
for j = 1 : m
 for k = 1 : t
  a(k, j + 1) = g^(j−1);
 end
end
for j = 1 : t
 for k = 1 : m
  b(k + 1, j) = g^(k−1);
```

```
end
end
L1 = kron((0 : m)', ones(t, 1)); % 基本列第一列
L2 = kron(ones(t, 1), (0 : m)'); % 基本列第二列
L = [L1, L2, ones(n, m)];
i = 1;
while(i <= m)
B = mod(a + i * b, t); %正交拉丁方完全组
H = reshate(B', n, 1); % 正交拉丁方完全组按行展开
L(:, 2 + i) = L(:, 2 + i, :). * H; % .* 表示对应位置的元素相乘
i = i + 1;
end
L
```

在$Matlab$程序中只需改变初始条件t的值, 我们就构造$L_{t^2}(t^m)$ 型正交表.

引理2.2.5. *([17])* 设R_P为以素数p为模的剩余类域, 则元素对模p的加法与乘法来说, 所有方阵

$$C_i = \begin{pmatrix} 1 & 2 & \cdots & p \\ 1+i & 2+i & \cdots & p+i \\ 1+2i & 2+2i & \cdots & p+2i \\ \cdots & \cdots & \cdots & \cdots \\ 1+(p-1)i & 2+(p-1)i & \cdots & p+(p-1)i \end{pmatrix},$$

其中$i = 1, 2, \cdots, p-1$, 构成p阶正交拉丁方完全组.

第三种方法: 由引理2.2.5得到的正交拉丁方A_i按行展开, 添上基本列$(\underbrace{1, \cdots, 1}_{t}, \underbrace{2, \cdots, 2}_{t}, \cdots, \underbrace{t, \cdots, t}_{t})$和$(\underbrace{1, 2, \cdots, t}_{t}, \underbrace{1, 2, \cdots, t}_{t}, \cdots, \underbrace{1, 2, \cdots, t}_{t})$, 即可得到$L_{t^2}(t^m)$型正交表, 具体例子

如下:

例2.2.6. 以$L_{49}(7^8)$为例,此时$t=7$, 构造7阶正交拉丁方完全组, 从而得到正交表$L_{49}(7^8)$, 结果如表$A-6$, $Matlab$程序与解释如下:

```
clear;
t = 5;%初始条件
m = t − 1; n = t^2
for j = 1 : t
for k = 1 : t
a(k, j) = j;
b(k, j) = k − 1;
end
end
a, b
L1 = kron((0 : m)', ones(t, 1)); % 基本列第一列
L2 = kron(ones(t, 1), (0 : m)'); % 基本列第二列
L = [L1, L2. ones(n, m)];
i = 1;
while(i <= m)
B = mod(a + i * b, t); %正交拉丁方完全组
H = reshate(B', n, 1); % 正交拉丁方完全组按行展开
L(:, 2 + i) = L(:, 2 + i, :). * H; % .* 表示对应位置的元素相乘
i = i + 1;
end
L
```

在$Matlab$程序中只需改变初始条件t的值，我们就构造$L_{t^2}(t^m)$ 型正交表.

§2.3 $L_{\lambda p^2}(p^{\lambda p+1})$型正交表

引理2.3.1. *([17])* 设$D(\lambda,p)$为p阶加群G的一个差集合，若把$D(\lambda,p)$中每个元素换成G的加法表中所对应的列，则由此得到$\lambda p^2\times\lambda p$矩阵是一个正交表$L_{\lambda p^2}(p^{\lambda p})$；并且这张表是可分解的，就是说，它的$\lambda p^2$行可以分成$\lambda p$ 组，每组包含p 行，而任一列中每组的p个元素都正好是G的全体元素. 因此，可在$L_{\lambda p^2}(p^{\lambda p})$中再添上如下一列：

$$\{0,\cdots,0,1,\cdots,1,\cdots,p-1,\cdots,p-1\}'$$

便得到一个正交表$L_{\lambda p^2}(p^{\lambda p+1})$.

由引理2.3.1可知，以差集合为基础可以构造$L_{\lambda p^2}(p^{\lambda p+1})$ 型正交表. 本章节中p是素数，设$R_P=\{0,1,2,\cdots,p-1\}$是以p为模的剩余类域，针对以下三种情形分别介绍构造差集合$D(\lambda,p)$的方法:

情形一: 当素数$p=6n-1$时，如文献[17]所示，构造差集合$D(\lambda,p)$，此时$\lambda=2$，作以下四个R_P上的p阶方阵：

1. $A=(a_{ij})$，其中$a_{ij}=ij$;
2. $B=(b_{ij})$，其中$b_{ij}=i(i+j)$;
3. $C=(c_{ij})$，其中$c_{ij}=(i+j)j$;
4. $D=(d_{ij})$，其中$d_{ij}=-3^{-1}(i^2+ij+j^2)$，$-3^{-1}$ 指的是在域R_P 中3的逆元的负元;

则可得到一个差集合$D(2,p)=\begin{pmatrix}\mathrm{A} & \mathrm{B}\\ \mathrm{C} & \mathrm{D}\end{pmatrix}$，其中i, j=0, 1, 2, $\cdots$, $p-1$，而且以上的加与乘都是p 为模的加与乘.

例2.3.2. 以正交表$L_{50}(5^{11})$为例，则素数$p=6n-1=5$，即$n=1$，构造差集合$D(2,5)$，再将$D(2,5)$中每一个元素换成5阶加群的加法表G所对应的列. 则由此得到$L_{50}(5^{10})$，再添加如下一列：$(\underbrace{0,\cdots,0}_{2p},\underbrace{1,\cdots,1}_{2p},\cdots,\underbrace{p-1,\cdots,p-1}_{2p})$，便得到正交表$L_{50}(5^{11})$，结果如表$A-7$，$Matlab$程序与解释如下:

```
clear;
n = 1 %初始条件
p = 6 * n - 1, m = p - 1
for j = 0 : m
 for i = 0 : m
  G1(i + 1, j + 1) = mod(i * j, p); %乘法表
  G2(i + 1, j + 1) = mod(i + j, p); %加法表
 end
end
G1; G2;
x = find(G1(3 + 1, :) == 1) - 1; %3的逆元
y = find(G2(x + 1, :) == 0) - 1; %3的逆元的负元
for j = 0 : m
 for i = 0 : m
A(i + 1, j + 1) = mod(i. * j, p);
B(i + 1, j + 1) = mod(i. * (i + j), p);
C(i + 1, j + 1) = mod((i + j). * j, p);
D(i + 1, j + 1) = mod(y * (i. * i + i. * j + j. * j), p);
 end
end
A; B; C; D;
d = [A B; C D]; %差集合D(2,5)
for i = 1 : p
```

$d(d == (p-i)) = p-(i-1)$; %0, 1, $\cdots$, $p-1$用1, $\cdots$, $p-1$, p替换

end

$I = ones(p,1)$; $d = kron(d, I)$;

$i = 0$;

$while(i <= 2*m)$

$for\ j = 1:p$

$z = find(d(1+p*i,:) == j)$;

$[s\ t] = size(z)$;

$g = repmat(G2(:,j), 1, t)$;

$d((1+p*i):(p*(i+1)), z) = (d((1+p*i):(p*(i+1)), z) - (j-1)).*g$; % 差集合中的元素用加法表中相应的列替换

end

$i = i+1$;

end

d;

$L1 = d$ %正交表$L_{50}(5^{10})$

$L2 = [kron([0:m]', ones(2*p,1)), L1]$ %正交表$L_{50}(5^{11})$

在$Matlab$程序中只需改变初始条件n的值, 即可构造$L_{\lambda p^2}(p^{\lambda p+1})$ 型正交表.

情形二: 当素数$p = 6n+1 \not\equiv 1, 4(mod 5)$时, 如文献[17]所示, 构造差集合$D(\lambda, p)$, 此时$\lambda = 2$, 作以下四个$R_P$ 上的p阶方阵:

$A = (a_{ij})$, $a_{ij} = ij$;

$B = (b_{ij})$, $b_{ij} = i(j-i)$;

$C = (c_{ij})$, $c_{ij} = (i+j)j$;

$D = (d_{ij})$, $d_{ij} = 5^{-1}(-i^2+ij+j^2)$, 5^{-1} 指的是在域R_P中5的逆元;

则可得到一个差集合$D(2,p)=\begin{pmatrix} A & B \\ C & D \end{pmatrix}$, 其中$i,j=0,1,2,\cdots,p-1$, 而且以上的加与乘都是$p$为模的加与乘.

例2.3.3. 以正交表$L_{98}(7^{15})$为例, 则素数$p=6n+1=7\not\equiv 1,4(mod5)$, 即$n=1$, 构造差集合$D(2,7)$, 再将$D(2,7)$中每一个元素换成7阶加群的加法表$G$所对应的列. 则由此得到$L_{98}(7^{14})$, 再添加如下一列: $(\underbrace{0,\cdots,0}_{2p},\underbrace{1,\cdots,1}_{2p},\cdots,\underbrace{p-1,\cdots,p-1}_{2p})$, 便得到正交表$L_{98}(7^{15})$, 结果如表$A-8$, $Matlab$程序与解释如下:

```
clear;
n = 1 %初始条件
p = 6*n+1, m = p-1
for j = 0:m
 for i = 0:m
  G1(i+1,j+1) = mod(i*j,p); %乘法表
  G2(i+1,j+1) = mod(i+j,p); %加法表
 end
end
G1; G2;
x = find(G1(5+1,:)==1)-1 %5的逆元
for j = 0:m
for i = 0:m
A(i+1,j+1) = mod(i.*j,p);
B(i+1,j+1) = mod(i.*(j-i),p);
C(i+1,j+1) = mod((i+j).*j,p);
D(i+1,j+1) = mod(x*(-i.*i+i.*j+j.*j),p);
```

end

end

$A;\ B;\ C;\ D;$

$d = [A\ B; C\ D];$ %差集合$D(2,7)$

$for\ i = 1 : p$

$d(d == (p-i)) = p-(i-1);$ %$0,1,\cdots,p-1$用$1,\cdots,p-1,p$ 替换

end

$I = ones(p,1);\ d = kron(d,I);$

$i = 0;$

$while(i <= 2*m)$

$for\ j = 1 : p$

$z = find(d(1+p*i,:) == j);$

$[s\ t] = size(z);$

$g = repmat(G2(:,j),1,t);$

$d((1+p*i):(p*(i+1)),z) = (d((1+p*i):(p*(i+1)),z)-(j-1)).*g;$ %差集合中的元素用加法表中相应的列替换

end

$i = i+1;$

end

$d;$

$L1 = d$ %正交表$L_{50}(5^{10})$

$L2 = [kron([0:m]', ones(2*p,1)), L1]$ %正交表$L_{50}(5^{11})$

在$Matlab$程序中只需改变初始条件n的值, 我们就构造$L_{\lambda p^2}(p^{\lambda p+1})$型正交表.

情形三: 当素数$p = 6n + 1 \neq 1, 2, 4(mod7)$, 且$6n + 1 \neq 7$时, 如文献[17]所示, 构造差集合$D(\lambda, p)$, 此时$\lambda = 2$, 作以下四个$R_P$上的$p$ 阶方阵:

$A = (a_{ij})$, $a_{ij} = ij$;

$B = (b_{ij})$, $b_{ij} = i(j - i)$;

$C = (c_{ij})$, $c_{ij} = (i - 2j)j$;

$D = (d_{ij})$, $d_{ij} = 7^{-1}(i^2 - ij + 2j^2)$, 7^{-1} 指的是在域R_P 中7的逆元;

则可得到一个差集合$D(2,p)=\begin{pmatrix} A & B \\ C & D \end{pmatrix}$, 其中$i$, j=0, 1, 2, $\cdots$, $p-1$, 而且以上的加与乘都是p为模的加与乘.

例2.3.4. 以正交表$L_{338}(13^{27})$ 为例, 则素数$p = 6n + 1 = 13 \neq 1, 2, 4(mod7)$, 即$n = 2$, 构造差集合$D(2, 13)$, 再将$D(2, 13)$中每一个元素换成13阶加群的加法表$G$ 所对应的列. 则由此得到$L_{338}(13^{26})$, 再添加如下一列: $(\underbrace{0, \cdots, 0}_{2p}, \underbrace{1, \cdots, 1}_{2p}, \cdots, \underbrace{p-1, \cdots, p-1}_{2p})$, 便得到一个正交表$L_{338}(13^{27})$, 为了节省篇幅, 表格省略, $Matlab$ 程序与解释如下:

```
clear;
n = 2, p = 6 * n + 1, m = p - 1 %初始条件
for j = 0 : m
for i = 0 : m
G1(i + 1, j + 1) = mod(i * j, p); %乘法表
G2(i + 1, j + 1) = mod(i + j, p); %加法表
end
end
G1, G2
x = find(G1(7 + 1, :) == 1) - 1 %7的逆元
for j = 0 : m
```

$for\ i = 0 : m$

$A(i+1, j+1) = mod(i. * j, p);$

$B(i+1, j+1) = mod(i. * (j-i), p);$

$C(i+1, j+1) = mod((i - 2 * j). * j, p);$

$D(i+1, j+1) = mod(x * (i. * i - i. * j + 2 * j. * j), p);$

end

end

A, B, C, D

$d = [A\ B; C\ D]$ % 差集合$D(2, 13)$

$for\ i = 1 : p$

$d(d == (p-i)) = p - (i-1);$ %$0, 1, \cdots, p-1$用$1, \cdots, p-1, p$替换

end

$I = ones(p, 1);\ d = kron(d, I);$

$i = 0;$

$while(i <= 2 * m)$

$for\ j = 1 : p$

$z = find(d(1 + p * i, :) == j);$

$[s\ t] = size(z);$

$g = repmat(G2(:, j), 1, t);$

$d((1 + p * i) : (p * (i+1)), z) = (d((1 + p * i) : (p * (i+1)), z) - (j-1)). * g;$ %差集合中的元素用加法表中相应的列替换

end

$i = i + 1;$

end

d;

$L1 = d$ %正交表$L_{50}(5^{10})$

$L2 = [kron([0:m]', ones(2*p, 1)), L1]$ %正交表$L_{50}(5^{11})$

在此$Matlab$程序中只需改变初始条件n的值, 我们就构造$L_{\lambda p^2}(p^{\lambda p+1})$型正交表.

§2.4 $L_{t^u}(t^m)$型正交表

本文主要采用向量作内积的方法生成$L_{t^u}(t^m)$型正交表, 如文献[17], 在本章节假设t为任意一个素数或者素数方幂, 而$GF(t)$为t 阶有限域, 令

$$V_u=\{\text{一切}u\text{维向量}(a_1, a_2, \cdots, a_u)\},\text{其中}a_i \in GF(t),\ \text{且}u > 1,$$

并且V_u对于向量的加法以及域中元素与向量的乘法, 作成域$GF(t)$上包含t^u个向量的一个向量空间V_u. 在V_u中, 以下的$\frac{t^u-1}{t-1}$个向量就是V_u的全部标准向量:

$(1,$	$0,$	$0,$	$\cdots,$	$0,$	$0,$	$0)$	1 个
$(b_{u-1,1},$	$1,$	$0,$	$\cdots,$	$0,$	$0,$	$0)$	t个
$(b_{u-2,1},$	$b_{u-2,2},$	$1,$	$\cdots,$	$0,$	$0,$	$0)$	t^2个
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$(b_{2,1},$	$b_{2,2},$	$b_{2,3},$	$\cdots,$	$b_{2,u-2},$	$1,$	$0)$	t^{u-2}个
$(b_{1,1},$	$b_{1,2},$	$b_{1,3},$	$\cdots,$	$b_{1,u-2},$	$b_{1,u-1},$	$1)$	t^{u-1}个

其中每个b_{ij}为域$GF(t)$的所有元素, 共有t种取法.

构造$L_{t^u}(t^m)$型正交表的方法和步骤如下:

第1步, 给出t阶有限域$GF(t)$;

第2步, 给出域$GF(t)$上的全体u维向量和全部标准向量;

第3步, 以全体向量为行号, 以全体标准向量为列号, 做出所有内积即可得$L_{t^u}(t^m)$ 型正交表.

例2.4.1. 以$L_8(2^7)$为例,则$t=2,u=3$, 先构造一切3维向量V_3, 再构造$\frac{2^3-1}{2-1}$个标准向量, 作内积, 从而生成正交表$L_8(2^7)$, 结果如表$A-1$, $Matlab$程序与解释如下:

$clc; clear$

$t=2; u=3;$ %初始条件

$p=t; V=[\,]; K=[\,]; L=[\,]; D=[\,]; L1=[\,]; L2=[\,]; D=[\,];$

$G=transpose(0:(t-1));$ %t阶有限域

$i=1;$

$while(i<=u)$

$R=kron(kron(ones(t^{(i-1)},1),G),ones(t^{(u-i)},1));$

$V=[V,R];$

$i=i+1;$

end

V %所有的u维向量

$U=rref(V');$ %标准化

$[m,n]=find(U==1);$

$[s,t]=find(U((m+1):u,n)<=1);$

$B=U(:,unique(n(t)))';$ %标准向量

$L=V*B';$

$L=mod(L,p)$ %得到正交表$L_{t^u}(t^m)$

在$Matlab$程序中只需修改初始条件t, u的值, 我们就可以生成形如$L_{t^u}(t^m)$的正交表.

§2.5 混合水平正交表

混合水平正交表可由$L_{t^u}(t^m)$型正交表通过并列而产生, 如文献[17], 构造混合水平正交表的方法和步骤如下:

第1步, 可借助本文2.4的方法得到$L_{t^u}(t^m)$型正交表;

第2步, 从$L_{t^u}(t^m)$中任选k个独立列, 其中$k < u$;

第3步, 从$L_{t^u}(t^m)$中去掉所有可以用这k列线性表示的列, 设剩下r 列, 其中$r = m - \frac{t^k-1}{t-1}$, 构成正交表$L_{t^u}(t^r)$;

第4步, 将这k列同行元素所构成的t^k个互异的有序组, 按字典排列法与t^k个自然数1, 2, $\cdots$, t^k建立一一对应, 而把这k列所构成的每个有序组均换成与它对应的自然数, 便得到一个新的t^k水平列;

第5步, 在$L_{t^u}(t^r)$上添加这个新列, 即得到混合水平正交表$L_{t^u}(t^k \times t^r)$, $t_1 = t^k$.

由本文第二章第四节的程序得到$L_{t^u}(t^m)$型正交表, 取定k个独立的列($k < u$), 去掉能用这k列线性表示的所有列, 再将这k列所构成的t^k个互异的有序组, 把这k列所构成的每个有序组均换成它对应的自然数, 得到混合水平正交表$L_{t^u}(t_1 \times t^r)$, $t_1 = t^k$.

例2.5.1. 以$L_8(4 \times 2^3)$为例,则$2^3 = 8$, 此时$t = 2, u = 3, k = 2$, 由本文第二章第四节的程序得到$L_8(2^7)$, 取定2个独立的列($2 < 3$), 去掉能用这2列线性表示的所有列, 再将这2列所构成的$2^2 = 4$个互异的有序组, 把这2列所构成的每个有序组均换成它对应的自然数, 从而生成正交表$L_8(4 \times 2^3)$, 结果如表$A-8$, $Matlab$ 程序与解释如下:

```
clc; clear
t = 2; u = 3; k = 3; %初始条件
p = t; [z, z1] = size(L);
K = L(:, 1 : k); %默认1 : k列, 也可以换成其他列
i = 1;
while(i <= z1)
d = rank([K, L(:, i)]') %m个n维向量线性相关等价于秩< m
D = [D, d];
i = i + 1;
end
```

D

$a = find(D(1,:) >= (k+1))$

$L1 = L(:,a)$ %得到正交表$L_{t^u}(t^r)$

%以下生成混合水平列

$Y = unique(K,'rows')$ %删除K中重复的行, 得到有序组

$[y, y1] = size(Y);$

$j = 1;$

$while(j <= y)$

$b = find(K(:,1) == Y(j,1) \& K(:,2) == Y(j,2));$

$[z, z1] = size(b);$

$K(b,:) = (y-(j-1)) * ones(z,k);$ %有序组与自然数建立一一对应

$j = j+1;$

end

$L2 = [flipud(K(:,1)), L1]$%得到混合水平正交设计表

在$Matlab$ 程序中只需修改初始条件t, u, k的值, 我们就可以生成形如$L_{t^u}(t_1 \times t^r)$, $t_1 = t^k$的混合水平正交表.

§2.6 高强度正交表

引理2.6.1. *([5])* 设M是一个选自Q^n的任意$t(1 \le t \le q^{n-1})$个行向量构成的矩阵, 则

$$L = \begin{pmatrix} 0 \oplus M \\ 1 \oplus M \\ 2 \oplus M \\ \cdots\cdots \\ q-1 \oplus M \end{pmatrix}$$

是一个强度为1的正交表.

引理2.6.2. *([5])* 对于任意的整数$1 \le k \le q$, 总存在Q^n 上的1−正交分划$A_1, A_2, \cdots, A_k$.

证明: 由于Q^n本身就是一个正交表, 所以当$k = 1$时定理显然成立, 当$k < q$ 时,可以将其中的某些A_i合并为一个, 使总数为k, 只需要证明当$k = q$时定理成立即可.

由于Q^n中有q^n个q元n维向量, 考虑将Q^n分为行数相等的分划容量为q的A_1, A_2, $\cdots$, A_k, 其中的每个A_i $(1 \le i \le k)$中都有q^{n-1}个向量, 为了得到A_1:

第*1*步, 从Q^n中任选q^{n-2}个向量, 组成矩阵A_1^0;

第*2*步, 由引理2.6.1可知:$0 \oplus A_1^0$、$1 \oplus A_1^0$、$\cdots$、$(q-1) \oplus A_1^0$构成一个强度1的正交表, 将其记为A_1;

第*3*步, 从$Q^n - A_1$中任选q^{n-2}个向量, 组成矩阵A_2^0;

第*4*步, 由引理2.6.1可知:$0 \oplus A_2^0$、$1 \oplus A_2^0$、$\cdots$、$(q-1) \oplus A_2^0$构成一个强度1的正交表, 将其记为A_2;

.....

第$2q-1$步, 将剩下的q^{n-1}个向量,组成矩阵A_q, 它当然是一个强度1的正交表.

这样就构造出了Q^n的一个1−正交分划$A_1, A_2, \cdots, A_q$.

引理2.6.3. *([5])* 若有Q^n上的正交分划$A_1, A_2, \cdots, A_q$, 如果$A_i (i = 1, 2, \cdots, q)$都是$q^{n-1}$行$n$ 列的强度为t的正交表, 那么下列矩阵都是强度为$t+1$的正交表, 并且是Q^{n+1} 的*(t+1)*-正交分划.

$$L_1 = \begin{pmatrix} 0 & A_1 \\ 1 & A_2 \\ 2 & A_3 \\ \vdots & \vdots \\ q-1 & A_q \end{pmatrix}, L_2 = \begin{pmatrix} 0 & A_q \\ 1 & A_1 \\ 2 & A_2 \\ \vdots & \vdots \\ q-1 & A_{q-1} \end{pmatrix}, \cdots, L_q = \begin{pmatrix} 0 & A_2 \\ 1 & A_3 \\ 2 & A_4 \\ \vdots & \vdots \\ q-1 & A_1 \end{pmatrix},$$

其中的0, 1, 2, $\cdots$, $q-1$均指与其对应的A_i有相同的行的元素全为0, 1, 2, $\cdots$, $q-1$的向量.

高强度正交表的构造可由引理2.6.1, 2.6.2得到强度为1的正交表, 和Q^n的1−正交分划, 在这个前提下, 再由引理2.6.3得到强度为2的正交表, 和Q^{n+1}的2− 正交分划, 接着由引理2.6.3得到强度为3的正交表, 和Q^{n+2}的3−正交分划, 依次下去, 逐步构造出高强度的正交表.

例2.6.4. 以$L_8(2^4)$为例,则$Q=\{0,1\}, q=2, N=4, t=3$, 首先生成强度为1的正交表,和的1−正交分划, 再生成强度为2的正交表, 和的2−正交分划, 接着生成强度为3的正交表, 和的3−正交分划, 这样就可以生成2水平强度为3的正交表. 结果如表$A-9$ 和$A-10$, $Matlab$程序与解释如下:

```
clc; clear;
q = 2; %水平(初始条件)
t = 3; %强度(初始条件)
N = 4; %列数(初始条件)
n = N − (t − 1); %q水平强度为1的正交表的列数
Q = [ ]; a0 = [ ]; a1 = [ ]; a = [ ]; A1 = [ ]; A = [ ]; L = [ ]; L0 = [ ]; L1 = [ ];
%第一部分: 生成Q^n即所有的q元n维向量
i = 1;
while(i <= n)
R = kron(kron(ones(q^(i−1), 1), [0 : (q − 1)]'), ones(q^(n−i), 1));
Q = [Q, R]; %Q^n即所有的q元n维向量
i = i + 1;
end
%第二部分: 生成1−正交分划
j = 1;
while(j <= (q − 1))
for i = 0 : (q − 1);
```

$a0 = Q((1 + q^{n-2} * (j-1)) : q^{n-2} * j, :);$

$a1 = mod(i + a0, q);$ %正交分划

$a = [a; a1];$

end

$Q = setdiff(Q, repmat(a, 2, 1), 'rows');$ %$Q^n - a$

$j = j + 1;$

end

$A = [a; Q];$ %生成强度为1的正交表

%第三部分：正交分划的递归构造

$while(n <= N - 1)$ %Q正交分划$q^{(n-2)}$行向量的个数

$L0 = kron(ones(q, 1), kron([0 : q-1]', ones(q^{n-1}, 1)));$

$j = 0;$

$while(j <= q - 1)$

$A1 = circshift(A, [q^{n-1} * j, 0]);$ %每次向下移q^{n-1}

$L1 = [L1; A1];$

$j = j + 1;$

end

$A = [L0, L1];$ %所有正交分划的并列

$L1 = [\,];$

$n = n + 1;$

end

%第四部分：生成强度为t的正交表

$for\ i = 1 : q;$

$L = A((1 + q^{n-1} * (i-1)) : q^{n-1} * i, :)$ %强度为t的正交表

end

在*Matlab*程序中, 保证$n = N - (t-1) >= 2$的前提下, 只需修改初始条件q, N, t 的值, 我们就生成q水平, 强度为t, 列数为N, 行数为q^{N-1}的正交表. 比如(1)修改初始条件, 使得$q = 4$, $t = 3$, $N = 4$ 便可得到水平为4, 强度为3, 列数为4, 行数为64的正交表; (2)修改初始条件, 使得$q = 5$, $t = 3$, $N = 4$, 便可得到水平为5, 强度为3, 列数为6, 行数为125的正交表; (3)修改初始条件, 使得$q = 7$, $t = 3$, $N = 4$便可得到水平为7, 强度为3, 列数为6, 行数为343 的正交表; (4)修改初始条件, 使得$q = 8$, $t = 3$, $N = 4$, 便可得到水平为8, 强度为3, 列数为6, 行数为512的正交表.

第三章　正交表在数值计算的应用

本章中给出了求解非线性方程组和超定方程组的方法, 具体情况如下：

§3.1　问题转化方法与原理

对于方程组

$$\begin{cases} F_1(t_1, t_2, \cdots, t_m) = g_1 \\ F_2(t_1, t_2, \cdots, t_m) = g_2 \\ \qquad\qquad \vdots \\ F_s(t_1, t_2, \cdots, t_m) = g_s \end{cases} \tag{3-1}$$

其中$F_1, F_2, \cdots, F_s$为已知的多元函数, 其中自变量的定义域为: $t_j \in (a_j, b_j), j = 1, 2, \cdots, m.$ 在实际问题中, 变量亦被称为因子. 要求解方程组$(3-1)$, 实际上就是求函数

$$F(t_1, t_2, \cdots, t_m) = \sum_{k=1}^{s} (F_k(t_1, t_2, \cdots, t_m) - g_k)^2 \tag{3-2}$$

的最小值, 其中$t_j \in (a_j, b_j), j = 1, 2, \cdots, m.$

利用一定的变换将函数的定义域$\prod_{j=1}^{m}(a_j, b_j)$转换到$\prod_{j=1}^{m}(0, 1)$, 利用文献[3], 在通常情况下可分为以下四种情况:

1. 若a_j, b_j都是有限数, 可取$t_j = a_j + x_j(b_j - a_j)$;

2. 若a_j为有限数, b_j为$+\infty$, 可取$t_j = a_j + \frac{x_j}{1-x_j}$;

3. 若a_j为$-\infty$, b_j有限数, 可取$t_j = \ln(\frac{x_j}{c_j - x_j})$, $c_j = \frac{1+e^{b_j}}{e^{b_j}}$;

4. 若a_j为$-\infty$, b_j为$+\infty$, 可取$t_j = \ln\frac{1-x_j}{x_j}$;

其中$x_j \in (0, 1), j = 1, 2, \cdots, m.$

于是可将函数$(3-2)$变成

$$f(x_1, x_2, \cdots, x_m) = \sum_{k=1}^{s} (f_k(x_1, x_2, \cdots, x_m) - g_k)^2 \tag{3-3}$$

其中$x_j \in (0,1)$, $j=1,2,\cdots,m$.

接下来就是要在多维区间$\prod_{j=1}^{m}(0,1)$内来寻找函数$(3-3)$的最小值, 可利用区间收缩法求函数极小值进而求解方程组, 如文献[19], 具体步骤如下:

1. 首先需选用合适的正交表, 根据正交试验的表头设计安排一个试验. 设H为一个正交设计,

$$H=(a_{ij})_{n\times m}, 0\le a_{ij}\le p_j-1,\ a_{ij}\text{是整数},\ i=1,2,\cdots,n;\ j=1,2,\cdots,m,$$

其中p_j为H的第j列的水平数, p_j可以相等, 也可以不相等;

2. 根据水平变换$x_{ij}=a_j+\frac{(2a_{ij}+1)(b_j-a_j)}{4}$, 得到$x_i^T=(x_{i1},x_{i2},\cdots,x_{im})$ 为H的第i行$(a_{i1},a_{i2},\cdots,a_{im})$对应的试验点, 这样就把$H$转化为矩阵$X=(x_{ij})_{n\times m}$;

3. 将$x_i^T=(x_{i1},x_{i2},\cdots,x_{im})$代入函数式$(3-3)$中, 利用$Matlab$计算出$n$个相应的值$f(1)$, $f(2),\cdots,f(n)$, 进而得到相应的均值矩阵$\mu=(\mu_{jk})_{j\times k}=\frac{1}{r_{jk}}\sum_{i\in H_{jk}}f(i)$;

4. 查找出均值矩阵μ最大值在各因子的0水平或1水平取得, 若μ_{j0}最大, 则因子x_j所在区间的左端点增加区间长度的1/4; 若μ_{j1}最大, 则因子x_j所在区间的右端点减小区间长度的1/4, 此时相应因子的区间长度收缩1/4个长度, 其他因子取值区间不变, 进而得到新的区间;

5. 重复步骤3、4, 直到各因子取值区间收缩至$|b_j-a_j|\le 0.0001$, 即可得到函数式$(3-2)$的极小值点, 此过程通过$Matlab$ 实现;

6. 将其代入函数式$(3-2)$中验证结果, 若误差小, 就将这个极小值点作为方程组的解或者近似解; 若误差大, 这个极小值点虽然不是方程组的解, 但是我们可以判断解就在这个极小值点附近, 不过取值范围有可能偏大也可能偏小, 所以有2^m种可能, 此时可以确认新的取值范围;

7. 在新的取值范围内, 对函数$(3-2)$进行步骤1, 2, 3, 4, 进而求得函数$(3-2)$的极小值点, 重新代入方程组验证, 误差小的则可直接作为方程组的解或者近似解, 此时的解会更精确, 甚至能求到方程组的多个解.

由文[19]可知, 此正交试验是完全试验, 所以可以认为我们求解的方程组的方法是正确. 这种方法得到的解精确度高, 且大大缩减了复杂方程组的计算量, 用时少, 不需要初始值.

§3.2 实际应用

为了说明方法的可行性, 下面我们将验证一些比较简单的例子, 本文所有计算结果都是在以下环境下完成的, 处理器: $Intel(R)\ Core(TM)i3-23-2310M\ CPU\ @\ 2.10GHz$ $2.10GHz$, 内存: $2.00GB$, 操作系统: $Windows10$, 运行系统: $MatlabR2012a$.

§3.2.1 一类非线性方程组的求解

例3.2.1. 求解如下2元非线性方程组的解:

$$\begin{cases} 4t_1^2+t_2^2+2t_1t_2-t_2=2 \\ 2t_1^2+3t_1t_2+t_2^2=3 \end{cases} \tag{3-4}$$

解 步骤如下:

1. 构造函数

$$F(t_1,t_2)=(4t_1^2+t_2^2+2t_1t_2-t_2-2)^2+(2t_1^2+3t_1t_2+t_2^2-3)^2; \tag{3-5}$$

2. 因为$t_j\in(-\infty,\infty)$, 所以令$t_j=\ln(\frac{1-x_j}{x_j})$, 其中$x_j\in(0,1)$, $j=1,2$, 可将$(3-5)$式改写成

$$\begin{aligned} f(x_1,x_2)=&(4\ln^2\frac{1-x_1}{x_1}+\ln^2\frac{1-x_2}{x_2}+2\ln\frac{1-x_1}{x_1}\ln\frac{1-x_2}{x_2}-\ln\frac{1-x_2}{x_2}-2)^2 \\ &+(2\ln^2\frac{1-x_1}{x_1}+3\ln\frac{1-x_1}{x_1}\ln\frac{1-x_2}{x_2}+\ln^2\frac{1-x_2}{x_2}-3)^2; \end{aligned} \tag{3-6}$$

3. 根据正交试验表头设计将x_1,x_2安排在正交表$L_4(2^3)$的$1,2$两列, 如表$3-1$, 详见表$B-1$;

表3-1. 正交表$L_4(2^3)$表头设计

表头设计	x_1	x_2	
列号	1	2	3

4. 由变换$x_{ij}=a_j+\frac{(2a_{ij}+1)(b_j-a_j)}{4}$, $(i=1,2,3,4;\ j=1,2)$, 得到$x_i^T=(x_{i1},x_{i2})$为矩阵H的第i行(a_{i1},a_{i2})对应的试验点, 进而得到矩阵$X=(x_{ij})_{4\times 2}$, 其中

$$X=\begin{pmatrix} 0.3775 & 0.2689 \\ 0.3775 & 0.2690 \\ 0.3776 & 0.2689 \\ 0.3776 & 0.2690 \end{pmatrix};$$

5. 将$x_i^T=(x_{i1},x_{i2})$代入$(3-6)$中, 利用$Matlab$计算出4个相应的值$f(1),f(2),f(3),f(4)$, 进而得到相应的均值矩阵$\mu=(\mu_{jk})_{j\times k}=\frac{1}{r_{jk}}\sum\limits_{i\in H_{jk}}f(i)$, $k=0,1,j=1,2$, 其中

$$\mu=\begin{pmatrix} \mu_{10} & \mu_{20} \\ \mu_{11} & \mu_{21} \end{pmatrix}=1.0e-06\times\begin{pmatrix} 0.6546 & 0.6717 \\ 0.3663 & 0.3492 \end{pmatrix};$$

6. 收缩各因子的取值区间直至$|b_j-a_j|\leq 0.0001$, 就可以得到函数式$(3-5)$的极小值点$(0.50001,0.99996)^T$, 用时0.055103 秒, 此过程通过$Matlab$实现, 进而得到$F(t_1,t_2)=8.4998e-09$, 误差较小, 可以将这个极小值点作为方程组的近似解;

7. 当然, 我们还可以在其附近求得更精确的解, 令$t_1\in(0.49,1.49)$; $t_2\in(0.5,2)$; 对函数$(3-5)$进行步骤$3,4,5,6$, 求得函数$(3-5)$的极值点$(0.50000,1.00000)$, 用时0.029720秒, 进而得到$F(t_1,t_2)=0$, 可直接将其作为方程组的解;

9 当令$t_1\in(-1.5,0)$; $t_2\in(1.5,3)$; 对函数$(3-5)$进行步骤3, 4, 5, 6, 求得函数$(3-5)$的极小值点$(-0.42772,2.38679)^T$, 用时0.029681秒, 进而得到$F(t_1,t_2)=2.6775$, 可将其作为方程组的近似解.

由于我们的试验是完全试验, 所以可以认为我们求解方程组的方法是正确的, 并与文[15]中的结果做了比较, 表$3-2$给出了几种计算结果的比较.

表3 − 2. 本文算法与文[15]算法的比较

方法	t_1	t_2	$F(t_1,t_2)$
牛顿法	*0.50000*	*1.00000*	*0*
本文算法	*0.50000*	*1.00000*	*0*
	-0.42772	*2.38679*	*2.6775*
人工鱼群算法	*0.50001*	*0.99991*	*8.4620e-08*
	-0.42772	*2.38677*	*2.6777*
耦合神经网络算法	*0.49953*	*1.00430*	*1.9520e-04*

计算结果表明，与人工鱼群算法和耦合神经网络算法相比，本文算法求得的解的精确度更高，甚至比解析算法求得解还要好，且该方法用时少，不用考虑初始点.

例3.2.2. 求解如下2元非线性方程组的解:

$$\begin{cases} \cos(2t_1)-\cos(2t_2)-0.4=0 \\ 2(t_2-t_1)+\sin(2t_2)-\sin(2t_1)-1.2=0 \end{cases} \tag{3-7}$$

解 步骤如下:

1. 构造函数

$$F(t_1,t_2)=(\cos(2t_1)-\cos(2t_2)-0.4)^2+(2(t_2-t_1)+\sin(2t_2)-\sin(2t_1)-1.2)^2; \tag{3-8}$$

2. 因为$t_j\in(-\infty,\infty)$, 所以令$t_j=\ln(\frac{1-x_j}{x_j})$, 其中$x_j\in(0,1), j=1,2$, 于是将$(3-8)$式改写成

$$\begin{aligned} f(x_1,x_2)=&(\cos(2\ln(\frac{1-x_1}{x_1}))-\cos(2\ln(\frac{1-x_2}{x_2}))-0.4)^2+(2(\ln(\frac{1-x_2}{x_2})\\ &-\ln(\frac{1-x_1}{x_1}))+\sin(2\ln(\frac{1-x_2}{x_2}))-\sin(2\ln(\frac{1-x_1}{x_1}))-1.2)^2; \end{aligned} \tag{3-9}$$

3. 方法同例3.2.1, 用$Matlab$计算求得函数$(3-8)$的极小值点为$(0.15652,0.49336)^T$, 用时0.033903 秒, 将其代入函数式$(3-8)$进而得到$F(t_1,t_2)=3.2978e-09$, 直接作为方程组的近似解.

由于我们的试验是完全试验，所以可以认为我们求解方程组的方法是正确的，并与文[18]中的结果做了比较，表3－3给出了几种计算结果的比较.

表3－3. 本文算法与文[18]算法的比较

方法	t_1	t_2	$\cos(2t_1)-\cos(2t_2)-0.4$
本文算法	0.15652	0.49336	3.2978e-09
CGSO算法	0.15641	0.49331	5.1859e-08
牛顿法	0.15	0.49	2.2716e-04

计算结果表明，与牛顿法和CGSO算法相比，本文算法求得的解的精确度更高，且该方法用时少.

例3.2.3. 求解如下3元非线性方程组的解:

$$\begin{cases} 3t_1-\cos(t_2t_3)-0.5=0 \\ t_1^2-81(t_2+0.1)^2+\sin t_3+1.06=0 \\ \exp(-t_1t_2)+20t_3+\frac{10\pi-3}{3}=0 \end{cases} \tag{3-10}$$

解　步骤如下:

1. 构造函数

$$\begin{aligned} F(t_1,t_2,t_3)=&(3t_1-\cos(t_2t_3)-0.5)^2+(t_1^2-81(t_2+0.1)^2+\sin t_3+1.06)^2 \\ &+(\exp(-t_1t_2)+20t_3+\frac{10\pi-3}{3})^2; \end{aligned} \tag{3-11}$$

2. 因为$t_j\in(-\infty,\infty)$，所以令$t_j=\ln\frac{1-x_j}{x_j}$，其中$x_j\in(0,1),j=1,2,3$; 于是将$(3-11)$式改写成

$$\begin{aligned} f(x_1,x_2,x_3)=&(3\ln\frac{1-x_1}{x_1}-\cos(\ln\frac{1-x_2}{x_2}\ln\frac{1-x_3}{x_3})-0.5)^2 \\ &+(\ln^2\frac{1-x_1}{x_1}-81(\ln\frac{1-x_2}{x_2}+0.1)^2+\sin(\ln\frac{1-x_3}{x_3})+1.06)^2 \\ &+(\exp(-\ln\frac{1-x_1}{x_1}\ln\frac{1-x_2}{x_2})+20\ln\frac{1-x_3}{x_3}+\frac{10\pi-3}{3})^2; \end{aligned} \tag{3-12}$$

3. 根据正交试验表头设计将x_1, x_2, x_3安排在正交表$L_8(2^7)$的1, 2, 4三列, 如表$3-4$, 详见表$B-2$;

表$3-4$. 正交表$L_8(2^7)$表头设计

表头设计	x_1	x_2		x_3			
列号	1	2	3	4	5	6	7

4. 由变换$x_{ij}=a_j+\frac{(2a_{ij}+1)(b_j-a_j)}{4}$, $(i=1,2,\cdots,8;\ j=1,2,3)$, 得到$x_i^T=(x_{i1},x_{i2},x_{i3})$为矩阵$H$的第$i$行$(a_{i1},a_{i2},a_{i3})$对应的试验点, 进而得到矩阵$X=(x_{ij})_{8\times3}$, 其中

$$X=\begin{pmatrix} 0.25000 & 0.25000 & 0.25000 \\ 0.25000 & 0.25000 & 0.75000 \\ 0.25000 & 0.75000 & 0.25000 \\ 0.25000 & 0.75000 & 0.75000 \\ 0.75000 & 0.25000 & 0.25000 \\ 0.75000 & 0.25000 & 0.75000 \\ 0.75000 & 0.75000 & 0.25000 \\ 0.75000 & 0.75000 & 0.75000 \end{pmatrix};$$

5. 将$x_i^T=(x_{i1},x_{i2},x_{i3})$代入$(3-12)$中, 利用$Matlab$计算出8 个相应的值$f(1),\cdots,f(8)$, 进而得到相应的均值矩阵$\mu=(\mu_{jk})_{j\times k}=\frac{1}{r_{jk}}\sum\limits_{i\in H_{jk}}f(i)$, $k=0,1,j=1,2,3$, 其中

$$\mu=\begin{pmatrix} \mu_{10} & \mu_{20} & \mu_{30} \\ \mu_{11} & \mu_{21} & \mu_{31} \end{pmatrix}=1.0e+04\times\begin{pmatrix} 1.02110 & 1.36446 & 1.05413 \\ 1.02222 & 0.67886 & 0.98919 \end{pmatrix};$$

6. 收缩各因子的取值区间直至$|b_j-a_j|\le0.0001$, 就可以得到函数式$(3-11)$的极小值点$(0.50000,0.00000,-0.52360)^T$, 用时0.225534 秒, 此过程通过$Matlab$实现, 将其代入函数式$(3-11)$得到$F(t_1,t_2,t_3)=6.0079e-10$, 则可直接作为方程组的解.

由于我们的试验是完全试验, 所以可以认为我们求解方程组的方法是正确的, 并与文[22] 中的结果做了比较, 表$3-5$给出了两种计算结果的比较.

表3－5. 本文算法与文[22]算法的比较

方法	t_1	t_2	t_3
本文算法	0.50000	0.00000	-0.52360
迭代法	0.5000	0.0000	-0.5236

计算结果表明，用本文算法求得的方程组的解的精确度跟迭代法一样，都能逼近精确值，且该方法用时少，不用考虑初始点.

例3.2.4. 求解如下4元非线性方程组的解:

$$\begin{cases} t_2t_3+t_2t_4+t_3t_4=0 \\ t_1t_3+t_1t_4+t_3t_4=0 \\ t_1t_2+t_1t_4+t_2t_4=0 \\ t_1t_2+t_1t_3+t_2t_3=1 \end{cases} \tag{3-13}$$

解　步骤如下:

1. 构造函数

$$\begin{aligned} F(t_1,t_2,t_3,t_4)=&(t_2t_3+t_2t_4+t_3t_4)^2+(t_1t_3+t_1t_4+t_3t_4)^2 \\ &+(t_1t_2+t_1t_4+t_2t_4)^2+(t_1t_2+t_1t_3+t_2t_3-1)^2; \end{aligned} \tag{3-14}$$

2. 因为$t_j\in(-\infty,\infty)$，所以令$t_j=\ln\frac{1-x_j}{x_j}$，其中$x_j\in(0,1),j=1,2,3,4$，于是将$(3-14)$式改写成

$$\begin{aligned} f(x_1,x_2,x_3,x_4)=&(\ln\frac{1-x_2}{x_2}\ln\frac{1-x_3}{x_3}+\ln\frac{1-x_2}{x_2}\ln\frac{1-x_4}{x_4}+\ln\frac{1-x_3}{x_3}\ln\frac{1-x_4}{x_4})^2 \\ &+(\ln\frac{1-x_1}{x_1}\ln\frac{1-x_3}{x_3}+\ln\frac{1-x_1}{x_1}\ln\frac{1-x_4}{x_4}+\ln\frac{1-x_3}{x_3}\ln\frac{1-x_4}{x_4})^2 \\ &+(\ln\frac{1-x_1}{x_1}\ln\frac{1-x_2}{x_2}+\ln\frac{1-x_1}{x_1}\ln\frac{1-x_4}{x_4}+\ln\frac{1-x_2}{x_2}\ln\frac{1-x_4}{x_4})^2 \\ &+(\ln\frac{1-x_1}{x_1}\ln\frac{1-x_2}{x_2}+\ln\frac{1-x_1}{x_1}\ln\frac{1-x_3}{x_3}+\ln\frac{1-x_2}{x_2}\ln\frac{1-x_3}{x_3}-1)^2; \end{aligned} \tag{3-15}$$

3. 根据正交试验表头设计将x_1, x_2, x_3, x_4安排在正交表$L_8(2^7)$的$1, 2, 4, 7$四列，如表$3-6$，详见表$B-3$;

表3-6. 正交表$L_8(2^7)$表头设计

表头设计	x_1	x_2		x_3		x_4	
列号	1	2	3	4	5	6	7

4. 收缩各因子取值区间直至$|b_j - a_j| \leq 0.0001$就得到函数得到函数$(3-14)$的极小值点$(-0.59509, -0.45804, -0.45804, 0.25067)^T$，用时0.210160秒，此过程通过$Matlab$实现.

5. 将上述极值点代入函数式$(3-14)$，得到$F(t_1, t_2, t_3, t_4) = 0.0606$，有误差，不过不是很大，所以我们可以判断解就在这附近，极值点的取值范围调动比较小，不过取值范围有可能偏大也可能偏小，所以有2^4种可能，此时可以确认新的取值范围;

6. 令$t_1 \in (-0.6, 0.7)$, $t_2 \in (-0.5, 0.7)$, $t_3 \in (-0.5, 0.7)$, $t_4 \in (-0.4, 0.3)$，对函数$(3-14)$式进行步骤3, 4，求得函数$(3-14)$的极值点$(-0.57735, -0.57735, -0.57735, 0.28868)^T$，用时0.076168秒，进而得到$F(t_1, t_2, t_3, t_4) = 1.0087e-10$，则可直接作为方程组的解，此时的解会更精确;

7. 令$t_1 \in (-0.6, 0.7)$, $t_2 \in (-0.4, 0.7)$, $t_3 \in (-0.4, 0.7)$, $t_4 \in (-0.4, 0.2)$，对函数$(3-14)$式进行步骤3, 4，求得函数$(3-14)$的极值点$(0.57735, 0.57735, 0.57735, -0.28867)^T$，用时0.076168 秒，进而得到$F(t_1, t_2, t_3, t_4) = 1.0087e-10$，则可直接作为方程组的近似解.

由于我们的试验是完全试验，所以可以认为我们求解方程组的方法是正确的，并与文[22]中的结果做了比较，表$3-7$给出了两种计算结果的比较.

表$3-7$. 本文算法与文[22]算法的比较

方法	t_1	t_2	t_3	t_4
本文算法	-0.57735	-0.57735	-0.57735	0.28868
	0.57735	0.57735	0.57735	-0.28867
迭代法	-0.57735	-0.57735	-0.57735	0.28868

计算结果表明, 用本文算法求得的方程组的解的精确度跟迭代法一样, 都能逼近精确值, 且该方法用时少, 不用考虑初始点; 还能求得方程组(3 – 13)的另一组解.

§3.2.2 一类超定方程组的求解

定义3.2.5. *([8])* 对于方程组

$$\begin{cases} a_{11}t_1 + a_{12}t_2 + \cdots + a_{1n}t_n = b_1 \\ a_{21}t_1 + a_{22}t_2 + \cdots + a_{2n}t_n = b_2 \\ \qquad\qquad \vdots \\ a_{m1}t_1 + a_{m2}t_2 + \cdots + a_{mn}t_n = b_m \end{cases} \tag{3-16}$$

如果$m > n$, 那么就称之为超定方程组.

定义3.2.6. *([7])* 当$m > n$时, 对于方程组(3 – 16)可能无解, 这时, 可求出一组数值x_1^0, $x_2^0, \cdots, x_m^0$, 使得$\sum_{i=1}^{m}(a_{i1}t_1 + a_{i2}t_2 + \cdots + a_{in}t_n - b_i)^2$ 最小, 这样的一组数值称为方程组的最小二乘解, 可作为方程组(3 – 16)的解或者近似解.

由定义3.2.5和3.2.6可知, 方程组(3 – 16)在一般情况下无解, 只能求得其在最小二乘意义下的最优解, 即求$\min(F(t_1, t_2, \cdots, t_n))$, 其中

$$F(t_1, t_2, \cdots, t_n) = \sum_{i=1}^{n}(a_{i1}t_1 + a_{i2}t_2 + \cdots + a_{in}t_n - b_i)^2 \tag{3-17}$$

$t_j \in (a_j, b_j), j = 1, 2, \cdots, n.$ 此时可借助文献[20, 21]中正交表的数据分析理论来研究最优解问题, 接着利用文献[19]中正交表区间收缩法的思想, 进而求得函数$F(t_1, t_2, \cdots, t_n)$的极值点为方程组(3 – 16)的解或者近似解. 为了说明其在实际应用和数学方面所具有的价值, 特举几个简单的例子说明, 具体情况如下.

例3.2.7. 求解如下超定方程组:

$$\begin{cases} 2t_1 - t_2 = 1 \\ 8t_1 + 4t_2 = 0 \\ 2t_1 + t_2 = 1 \\ 7t_1 - t_2 = 8 \\ 4t_1 = 3 \end{cases} \tag{3-18}$$

其中t_1, t_2为未知数.

解 这是一个二元的超定方程组, 方程组的个数大于未知数的个数, 很明显此方程组只能求得在最小二乘意义下的最优解, 求解步骤如下:

首先, 构造函数

$$f(t_1,t_2)=(2t_1-t_2-1)^2+(8t_1+4t_2)^2+(2t_1+t_2-1)^2+(7t_1-t_2-8)^2+(4t_1-3)^2 \tag{3-19}$$

其次, 因为$t_j \in (-\infty,\infty)$, 所以令$t_j=\ln(\frac{1-x_j}{x_j})$, 其中$x_j \in (0,1), j=1,2$, 于是将(3-8)式改写成

$$\begin{aligned} f(x_1,x_2)=&(2\ln(\frac{1-x_1}{x_1})-\ln(\frac{1-x_2}{x_2})-1)^2+(8\ln(\frac{1-x_1}{x_1})+4\ln(\frac{1-x_2}{x_2}))^2\\ &+(2\ln(\frac{1-x_1}{x_1})+\ln(\frac{1-x_2}{x_2})-1)^2+(7\ln(\frac{1-x_1}{x_1})-\ln(\frac{1-x_2}{x_2})-8)^2\\ &+(4\ln(\frac{1-x_1}{x_1})-3)^2 \end{aligned} \tag{3-20}$$

接着, 方法同例3.2.1, 用$Matlab$计算求得函数(3-19)的极小值点为$(0.79272,-1.46411)^T$, 用时0.065865秒.

最后, 可求出最小二乘误差平方和$f=6.2113$, 与文献[6, 8]中的算法相比, 结果很接近, 说明此方法具有一定地应用价值, 可将$(0.79272,-1.46411)^T$作为方程组$(3-18)$的近似解, 表$3-8$给出了几种计算结果的比较.

表3-8. 本文算法与其他算法的比较

方法	t_1	t_2	f
本文算法	0.79272	-1.46411	6.2113
遗传算法	0.7927	-1.4641	6.2113
模拟退火算法	0.79264	-1.4638	6.2113

例3.2.8. *[23]* 求解如下超定方程组:

$$\begin{cases} t_1 = 1 \\ t_1 = 0 \\ t_1 + t_2 + 3t_3 = 1 \\ t_1 + t_2 + t_3 = 0 \end{cases} \tag{3-21}$$

其中t_1, t_2, t_3为未知数.

解　线性矛盾方程组在实际工程问题中经常出现, 很明显此方程组只能求得在最小二乘意义下的最优解, 求解步骤如下:

首先, 构造函数

$$f(t_1, t_2, t_3) = (t_1 - 1)^2 + (t_1)^2 + (t_1 + t_2 + 3t_3 - 1)^2 + (t_1 + t_2 + t_3)^2 \tag{3-22}$$

其次, 因为$t_j \in (-\infty, \infty)$, 所以令$t_j = \ln(\frac{1-x_j}{x_j})$, 其中$x_j \in (0,1), j = 1,2,3$, 于是将$(3-21)$ 式改写成

$$\begin{aligned} f(x_1, x_2, x_3) = &(\ln(\frac{1-x_1}{x_1}) - 1)^2 + (\ln(\frac{1-x_1}{x_1}))^2 + (\ln(\frac{1-x_1}{x_1}) + \ln(\frac{1-x_2}{x_2}) \\ &+3\ln(\frac{1-x_3}{x_3}) - 1)^2 + (\ln(\frac{1-x_1}{x_1}) + \ln(\frac{1-x_2}{x_2}) + \ln(\frac{1-x_3}{x_3}))^2 \end{aligned} \tag{3-23}$$

接着, 方法同例3.2.3, 用$Matlab$计算求得函数$(3-22)$的极小值点为$(0.50000, -1.00003, 0.50000)^T$, 用时0.106451 秒.

最后, 可求出最小二乘误差平方和$f = 0.5000$, 与文献*[23]*中算法相比, 结果一致, 说明此方法具有一定地应用价值, 可将$(0.50000, -1.00003, 0.50000)^T$作为方程组$(3-21)$的近似解, 表$3-9$给出了两种计算结果的比较.

表$3-9$. 本文算法与其他算法的比较

方法	t_1	t_2	t_3	f
本文算法	*0.50000*	*-1.00003*	*0.50000*	*0.5000*
$4R$算法	*0.5*	*-1*	*0.5*	*0.5000*

例3.2.9. *[10]* 求解如下超定方程组:

$$\begin{cases} 5t_1 - 2t_2 = 10 \\ -2t_1 + 5t_2 - 2t_3 = 4 \\ -2t_2 + 6t_3 - 2t_4 = 3 \\ -2t_3 + 6t_4 = 2 \\ 2t_3 - 2t_4 = 3 \end{cases} \tag{3-24}$$

其中t_1, t_2, t_3, t_4为未知数.

解 这是一个研究基于超定线性方程组模型, 从而计算变电站的入地电流的实际问题, 如文献*[10]*, 用于解决实际电力系统的$HVDC$地中直流的计算问题, 其中t_j对应着电流I_j, $j = 1, 2, 3, 4$. 很明显此时程组的个数大于未知数的个数,只能求得其在最小二乘意义下的最优解, 求解步骤如下:

首先, 构造函数

$$\begin{aligned} f(t_1, t_2, t_3, t_4) = (5t_1 - 2t_2 - 10)^2 + (-2t_1 + 5t_2 - 2t_3 - 2)^2 + (-2t_2 + 6t_3 \\ -2t_4 - 3)^2 + (-2t_3 + 6t_4 - 2)^2 + (2t_3 - 2t_4 - 3)^2 \end{aligned} \tag{3-25}$$

其次, 因为$t_j > 0$, 所以令$t_j = \frac{x_j}{1-x_j}$; 于是可将$(3-23)$变成

$$\begin{aligned} f(x_1, x_2, x_3, x_4) = (5\frac{x_1}{1-x_1} - 2\frac{x_2}{1-x_2} - 10)^2 + (-2\frac{x_1}{1-x_1} + 5\frac{x_2}{1-x_2} - 2\frac{x_3}{1-x_3} - 2)^2 \\ +(-2\frac{x_2}{1-x_2} + 6\frac{x_3}{1-x_3} - 2\frac{x_4}{1-x_4} - 3)^2 + (-2\frac{x_3}{1-x_3} + 6\frac{x_4}{1-x_4} - 2)^2 \\ +(2\frac{x_3}{1-x_3} - 2\frac{x_4}{1-x_4} - 3)^2 \end{aligned} \tag{3-26}$$

接着, 方法同例3.2.4, 用$Matlab$计算求得函数$(3-25)$的极小值点为$(3.13568, 2.80468, 1.78986, 0.88403)^T$, 用时0.099549 秒.

最后, 可求出最小二乘误差平方和$f = 1.6534$, 相应的各个入地电流$I_1 = 3.13568$, $I_2 = 2.80468$, $I_3 = 1.78986$, $I_4 = 0.88403$, 与文献*[10]*相比, 结果很接近, 在此特别说明文献*[10]*给出的计算结果有误, 说明此方法具有一定地应用价值, 可将$(3.13568, 2.80468,$

$1.78986, 0.88403)^T$ 作为方程组$(3-24)$的近似解，表$3-10$ 给出了两种计算结果的比较.

表$3-10$. 本文算法与其他算法的比较

方法	t_1	t_2	t_3	t_4	f
本文算法	*3.13568*	*2.80468*	*1.78986*	*0.88403*	*1.6534*
最小二乘法	*3.13563*	*2.80463*	*1.78983*	*0.88402*	*1.6534*

第四章　总结和建议

正交表研究已经吸引了众多行业的研究者的普遍关注. 本文首先是在原有的理论基础上使用计算机程序*Matlab*生成六种类型的正交表, 但是本文只给出了一种生成单一水平的高水平高强度正交表构造方法, 所以接下来还需要继续寻找混合水平的高水平高强度正交表的构造方法.

进一步研究如何利用现有的正交表解决非线性方程组的解或者近似解, 以及超定方程组的解或者最小二乘解.

参考文献

[1] 陈雪平. 混合水平正交表交互作用的研究[D]. 上海: 华东师范大学, 2009.

[2] 陈远方, 林曦晨, 徐利华, 汤洪秀, 汪宏晶, 尹平. 常用正交表的构造原理及SAS实现[J]. 中国卫生统计, 2012, 29(4): 470-474.

[3] 陈志琴, 张晓丽, 罗纯, 张应山, 陈雪平. 用正交表求多元函数积分[J]. 上海应用技术学院学报(自然科学版), 2010, 10(2): 119-123.

[4] Du J, Wen Q Y, Zhang J. New Construction of Symmetric Orthogonal Arrays of Strength t. IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences. 2013, 9(96):1901-1904.

[5] 杜蛟, 王守印, 温巧燕, 庞善起. 强度m的对称正交表的递归构造[J]. 应用数学学报;2012,35(2):232-244.

[6] 蒋冬初, 何飞, 向继文. 遗传算法求解函数优化问题的$Matlab$实现[J]. 吉首大学学报, 2005, 26(2): 98- 100.

[7] 贾文新, 郑玉歌. 关于最小二乘解及其几何解释[J]. 焦作矿业学院学报, 1993, 32(3): 92-95.

[8] 李宝家, 刘昊阳. 超定方程组的一种解法[J]. 沈阳工业大学学报, 2002, 24(1): 76-77.

[9] Jiang L, Yin J X. An approach of constructing mixed-level orthogonal arrays of strength $\geqq 3$. Science China(Mathematics). 2013, 6(56):1109-1115.

[10] 刘同同. 基于超定线性方程组模型的$HVDC$地电流计算方法研究[J]. 现代电力2011, 28(3): 47-50.

[11] 李学斌, 余小领, 李斌. 运用$Excel$进行正交表的构建[J]. 河南科技学院学报(自然科学版), 2008, 36(4): 110-113.

[12] 庞善起. 正交表的构造及其应用[M]. 成都: 电子科技大学出版社, 2004: 66-200.

[13] Shanqi Pang, Ying Wang, Jiao Du and Wenju Xu. Iterative Constructions of Orthogonal Arrays of Strength t And Orthogonal Partitions. IEICE Transactions on Fundamentals of Elec- tronics, E100-A No. 1 pp. 308-311, January 2017.

[14] Shanqi Pang, Yajuan Wang, Guangzhou Chen and Jiao Du. The Existence of a Class of Mixed Orthogonal Arrays. IEICE Transactions on Fundamentals of Electronics, E99-A No. 4 pp. 863-868, April 2016.

[15] 王冬冬, 周永权. 人工鱼群算法在求解非线性方程组中的应用[J]. 计算机应用研究, 2007, 24(6): 242-244.

[16] 夏林林, 吴开腾. 大范围求解非线性方程组的指数同伦法[J]. 计算数学, 2014, 36(2): 215-224.

[17] 杨子胥. 正交表的构造[D]. 济南: 山东人民出版社, 1978.

[18] 赵光伟, 周永权. 用改进的人工萤火虫群优化算法求解非线性方程组. 数学的实践与认识[J], 2016, 46(1): 176-186.

[19] 张晓丽, 陈志琴, 罗纯, 张应山, 陈雪平. 利用正交表区间收缩法求函数极值[J]. 上海应用技术学院学报(自然科学版), 2010, 10(2): 115-118.

[20] 张应山. 多边矩阵理论[M]. 北京: 中国统计出版社, 1993.

[21] 张应山. 正交表的数据分析及其构造[D]. 上海: 华东师范大学, 2006.

[22] 张旭, 檀结庆. 三步五阶迭代方法解非线性方程组[J]. 计算数学, 2013, 25(3): 297-304.

[23] 李月清. 线性最小二乘问题求解讨论[J]. 北京工业职业技术学院学报, 2013, 12(3): 49-54.

[24] 孙麟平. 解超定病态线性方程组问题[J]. 高等学校计算数学学报, 1981, 25(3): 194-202.

[25] 张成兴. 变异自适应混沌粒子群算法求解线性超定方程[J]. 计算机仿真, 2014, 25(3): 297-304.

[26] 张冰冰. 蚁群算法在控制系统中的应用研究[D]. 新疆: 新疆大学, 2013.

附录A:正交表

表A-1. 正交表$L_8(2^7)$

行号 \ 列号	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0
2	1	0	1	0	1	0	1
3	0	1	1	0	0	1	1
4	1	1	0	0	1	1	0
5	0	0	0	1	1	1	1
6	1	0	1	1	0	1	0
7	0	1	1	1	1	0	0
8	1	1	0	1	0	0	1

表A-2. 正交表$L_{12}(2^{11})$

行号＼列号	1	2	3	4	5	6	7	8	9	10	11
1	0	0	0	0	0	1	0	0	0	0	0
2	0	0	1	1	0	0	1	0	1	1	0
3	0	0	0	1	1	0	0	1	0	1	1
4	1	0	0	0	1	0	1	0	1	0	1
5	1	1	0	0	0	0	1	1	0	1	0
6	0	1	1	0	0	0	0	1	1	0	1
7	1	1	1	1	1	0	0	0	0	0	0
8	1	0	1	1	0	1	1	1	0	0	1
9	0	1	0	1	1	1	1	1	1	0	0
10	1	0	1	0	1	1	0	1	1	1	0
11	1	1	0	1	0	1	0	0	1	1	1
12	0	1	1	0	1	1	1	0	0	1	1

表A-3. 正交表$L_{20}(2^{19})$

行号＼列号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
3	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1
5	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	0
6	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	0	1
7	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	0	1	0
8	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	0	1	0	1
9	1	0	1	0	1	0	1	0	1	0	1	0	1	1	0	1	0	1	0
10	0	1	0	1	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1
11	1	0	1	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0
12	0	1	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1
13	1	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0
14	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1
15	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0
16	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1
17	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0
18	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
19	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
20	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

表A-4. 正交表$L_9(3^4)$

行号＼列号	1	2	3	4
1	0	0	0	0
2	0	1	1	1
3	0	2	2	2
4	1	0	1	2
5	1	1	2	0
6	1	2	0	1
7	2	0	2	1
8	2	1	0	2
9	2	2	1	0

表A-5. 正交表$L_{25}(5^6)$

行号＼列号	1	2	3	4	5	6
1	0	0	0	0	0	0
2	0	1	1	1	1	1
3	0	2	2	2	2	2
4	0	3	4	4	4	4
5	0	4	3	3	3	3
6	1	0	1	2	4	3
7	1	1	2	3	0	4
8	1	2	3	4	1	0
9	1	3	0	1	3	2
10	1	4	4	0	2	1
11	2	0	2	4	3	1
12	2	1	3	0	4	2
13	2	2	4	1	0	3
14	2	3	1	3	2	0
15	2	4	0	2	1	4
16	3	0	4	3	1	2
17	3	1	0	4	2	3
18	3	2	1	0	3	4
19	3	3	3	2	0	1
20	3	4	2	1	4	0
21	4	0	3	1	2	4
22	4	1	4	2	3	0
23	4	2	0	3	4	1
24	4	3	2	0	1	3
25	4	4	1	4	0	2

表A-6. 正交表$L_{49}(7^8)$

行号＼列号	1	2	3	4	5	6	7	8
1	0	0	1	1	1	1	1	1
2	0	1	2	2	2	2	2	2
3	0	2	3	3	3	3	3	3
4	0	3	4	4	4	4	4	4
5	0	4	5	5	5	5	5	5
6	0	5	6	6	6	6	6	6
7	0	6	0	0	0	0	0	0
8	1	0	2	3	4	5	6	0
9	1	1	3	4	5	6	0	1
10	1	2	4	5	6	0	1	2
11	1	3	5	6	0	1	2	3
12	1	4	6	0	1	2	3	4
13	1	5	0	1	2	3	4	5
14	1	6	1	2	3	4	5	6
15	2	0	3	5	0	2	4	6
16	2	1	4	6	1	3	5	0
17	2	2	5	0	2	4	6	1
18	2	3	6	1	3	5	0	2
19	2	4	0	2	4	6	1	3
20	2	5	1	3	5	0	2	4
21	2	6	2	4	6	1	3	5
22	3	0	4	0	3	6	2	5
23	3	1	5	1	4	0	3	6
24	3	2	6	2	5	1	4	0

表A-7. 正交表$L_{50}(5^{11})$

行号＼列号	1	2	3	4	5	6	7	8	9	10	11
1	0	0	0	0	0	0	0	0	0	0	0
2	0	1	1	1	1	1	1	1	1	1	1
3	0	2	2	2	2	2	2	2	2	2	2
4	0	3	3	3	3	3	3	3	3	3	3
5	0	4	4	4	4	4	4	4	4	4	4
6	0	0	1	2	3	4	1	2	3	4	0
7	0	1	2	3	4	0	2	3	4	0	1
8	0	2	3	4	0	1	3	4	0	1	2
9	0	3	4	0	1	2	4	0	1	2	3
10	0	4	0	1	2	3	0	1	2	3	4
11	1	0	2	4	1	3	4	1	3	0	2
12	1	1	3	0	2	4	0	2	4	1	3
13	1	2	4	1	3	0	1	3	0	2	4
14	1	3	0	2	4	1	2	4	1	3	0
15	1	4	1	3	0	2	3	0	2	4	1
16	1	0	3	1	4	2	4	2	0	3	1
17	1	1	4	2	0	3	0	3	1	4	2
18	1	2	0	3	1	4	1	4	2	0	3
19	1	3	1	4	2	0	2	0	3	1	4
20	1	4	2	0	3	1	3	1	4	2	0
21	2	0	4	3	2	1	1	0	4	3	2
22	2	1	0	4	3	2	2	1	0	4	3
23	2	2	1	0	4	3	3	2	1	0	4
24	2	3	2	1	0	4	4	3	2	1	0

续表A-7. 正交表$L_{50}(5^{11})$

行号＼列号	1	2	3	4	5	6	7	8	9	10	11
25	2	4	3	2	1	0	0	4	3	2	1
26	2	0	1	4	4	1	0	3	2	2	3
27	2	1	2	0	0	2	1	4	3	3	4
28	2	2	3	1	1	3	2	0	4	4	0
29	2	3	4	2	2	4	3	1	0	0	1
30	2	4	0	3	3	0	4	2	1	1	2
31	3	0	2	1	2	0	3	4	1	4	3
32	3	1	3	2	3	1	4	0	2	0	4
33	3	2	4	3	4	2	0	1	3	1	0
34	3	3	0	4	0	3	1	2	4	2	1
35	3	4	1	0	1	4	2	3	0	3	2
36	3	0	3	3	0	4	2	1	1	2	4
37	3	1	4	4	1	0	3	2	2	3	0
38	3	2	0	0	2	1	4	3	3	4	1
39	3	3	1	1	3	2	0	4	4	0	2
40	3	4	2	2	4	3	1	0	0	1	3
41	4	0	4	0	3	3	2	4	2	1	1
42	4	1	0	1	4	4	3	0	3	2	2
43	4	2	1	2	0	0	4	1	4	3	3
44	4	3	2	3	1	1	0	2	0	4	4
45	4	4	3	4	2	2	1	3	1	0	0
46	4	1	1	3	2	3	4	4	5	2	5
47	4	1	1	3	2	3	4	4	5	2	5
48	4	1	1	3	2	3	4	4	5	2	5
49	4	1	1	3	2	3	4	4	5	2	5

表A-8. 正交表$L_8(4\times 2^3)$

行号＼列号	1	2	3	4	5
1	1	0	0	0	0
2	1	1	1	1	1
3	2	0	0	1	1
4	2	1	1	0	0
5	3	0	1	0	1
6	3	1	0	1	0
7	4	0	1	1	0
8	4	1	0	0	1

表A-9. 正交表$L_8(2^4)$

行号＼列号	1	2	3	4
1	0	0	0	0
2	0	0	1	1
3	0	1	0	1
4	0	1	1	0
5	1	0	0	1
6	1	0	1	0
7	1	1	0	0
8	1	1	1	1

表A-10. 正交表$L_8(2^4)$

行号＼列号	1	2	3	4
1	0	0	0	1
2	0	0	1	0
3	0	1	0	0
4	0	1	1	1
5	1	0	0	0
6	1	0	1	1
7	1	1	0	1
8	1	1	1	0

附录B

表B-1. 正交表$L_4(2^3)$表头设计

表头设计	x_1	x_2	
列号	1	2	3
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	0

表B-2. 正交表$L_8(2^7)$表头设计

表头设计	x_1	x_2		x_3			
列号	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1
3	0	1	1	0	0	1	1
4	0	1	1	1	1	0	0
5	1	0	1	0	1	0	1
6	1	0	1	1	0	1	0
7	1	1	0	0	1	1	0
8	1	1	0	1	0	0	1

表B-3. 正交表$L_8(2^7)$表头设计

表头设计	x_1	x_2		x_3		x_4	
列号	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1
3	0	1	1	0	0	1	1
4	0	1	1	1	1	0	0
5	1	0	1	0	1	0	1
6	1	0	1	1	0	1	0
7	1	1	0	0	1	1	0
8	1	1	0	1	0	0	1

致谢

本文是申请河南师范大学理学硕士学位的学位论文, 是在导师庞善起教授的精心指导下于2017 年3月完成的. 在此, 我谨向帮助我顺利完成毕业论文的人们表示感谢.

在论文将要完成之时, 我的内心很是激动, 从开始选题到论文完成, 有很多可敬可爱的老师、学长学姐、同窗好友、以及家人们给我鼓励和帮助, 在这里请接受我诚挚的谢意!

首先, 我要向我的导师庞善起教授表示我最由衷的感谢, 感谢庞老师在我研究生学习期间对我的付出. 庞老师不仅教给我许多科研必备的理论基础, 还传授于我可贵的科研方法和精神, 为我今后在专业领域中的学习和发展奠定了坚实的基础. 庞老师常常在忙碌的工作之余, 抽出时间来给我们上课和指导. 他严谨的科学态度, 精益求精的工作作风, 诲人不倦的高尚师德, 严以律己、宽以待人的崇高风范, 朴实无华、平易近人的人格魅力深深地感染和激励着我. 从课题的选择到最终完成, 庞老师都始终给予我细心的指导和不懈的支持, 在此谨向庞老师致以诚挚的谢意和崇高的敬意。

同时, 我也感谢院系领导的关怀和支持, 感谢杜蛟老师的悉心教诲, 在此, 我向他们致以崇高的敬意. 我还要感谢我的同窗徐温菊和汪颖同学及各位师弟师妹, 谢谢他们在学习和生活上给予我的帮助和支持. 还有我的大学同学王华生, 无论是在程序编写还是数学理论知识都给我了很大的帮助, 在此一并表示我诚挚的谢意和最衷心的祝福.

鹿姗姗

2017年3月

攻读学位期间发表的学术论文目录

[1] 庞善起, 鹿姗姗. 正交表的构造方法及$Matlab$实现[J]. 中国卫生统计, 2017, 34(2): 364-367.

[2] 庞善起, 鹿姗姗. 一类非线性方程组的求解[J]. 数学的实践与认识, 录用.

独创性声明

本人郑重声明: 所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果. 尽我所知, 除了文中特别加以标注和致谢的地方外, 论文中不包含其他人已经发表或撰写的研究成果, 也不包含为获得河南师范大学或其他教育机构的学位或证书所使用过的材料. 与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意.

签名: 鹿姗姗　　日期: 2017年6月5日

关于论文使用授权的说明

本人完全了解河南师范大学有关保留、使用学位论文的规定, 即: 有权保留并向国家有关部门或机构送交论文的复印件和磁盘, 允许论文被查阅和借阅. 本人授权河南师范大学可以将学位论文的全部或部分内容编入有关数据库进行检索, 可以采用影印、缩印或扫描等复制手段保存、汇编学位论文. (保密的学位论文在解密后适用本授权书)

签名: 鹿姗姗　　导师签名:　　日期: 2017年6月5日