# 实验四 代数变换模型

## 实验目的

1、了解加密解密基本概念；

2、学习并掌握Hill密码加密解密原理；

3、通过案例学习掌握相关算法、模型及其应用。

## 基本概念

**数据加密：**是一门历史悠久的技术，指通过加密算法和加密密钥将明文转变为密文，而解密则是通过解密算法和解密密钥将密文恢复为明文；

**明文：**指原始的或未加密的数据；**密文：**明文加密后的格式，是加密算法的输出信息；**密钥：**数字、字母或特殊符号组成的字符串，用它控制数据加密、解密的过程；**加密算法：**加密所采用的变换方法；

**希尔密码（Hill Cipher）：**是运用基本矩阵论原理的分组替换密码，由Lester S. Hill在1929年发明的。每个字母当作26进制数字：A=0, B=1, C=2… 一串字母当成n维向量，跟一个n×n的矩阵相乘，再将得出的结果MOD 26。基本思想：将n个明文字母通过线性变换转换为n个密文字母。其重要意义是第一次在密码学中用到了代数方法（代数变换、模运算）。

## 实验内容

### 问题1

对明文“Mr Hill made this code.”，运用不同的密钥矩阵，实现Hill加密和解密过程。

#### 问题分析

分析给出明文，有26个英文字母（有大小写之分）以及空格和句号，有多种编码手段，这里采用ASCII码进行编码（Tab-1）：

**Table 1 ASCII Control Code Chart**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ASCII Control Code Chart | | | | | | | | | | | | |
|  | Char | Dec | Hex | Char | Dec | Hex | Char | Dec | Hex | **Char** | Dec | Hex |
| Ctrl*@* | NUL | 0 | 00 |  | 32 | 20 | @ | 64 | 40 | , | 96 | 60 |
| CtrlA | SOH | 1 | 01 | ! | 33 | 21 | A | 65 | 41 | a | 97 | 61 |
| CtrlB | STY X | 2 | 02 | " | 34 | 22 | B | *66* | 42 | b | 98 | 62 |
| CtrlC | EXT | 3 | 03 | # | 35 | 23 | C | 67 | 43 | c | 99 | 63 |
| CtrlD | EOT | 4 | 04 | $ | 36 | 24 | D | 68 | 44 | d | 100 | 64 |
| CtrlE | ENQ | *5* | 05 | % | 37 | 25 | E | 69 | 45 | e | 101 | 65 |
| CtrlF | ACK | 6 | 06 | *&* | 38 | 26 | F | 70 | 46 | f | 102 | 66 |
| CtrlG | BEL | 7 | 07 | s | 39 | 27 | G | 71 | 47 | g | 103 | 67 |
| CtrlH | BS | 8 | 08 | ( | 40 | 28 | H | 72 | 48 | *h* | 104 | 68 |
| CtrlI | HT | 9 | 09 | ) | 41 | 29 | I | 73 | 49 | j | 105 | 69 |
| CtrlJ | LF | 10 | 0A | \* | 42 | 2A | J | 74 | 4A | i | 106 | 6A |
| CtrlK | vr | 11 | 0B | + | 43 | 2B | K | 75 | 4B | k | 107 | 6B |
| CtrlL | FF | 12 | OC | , | 44 | 2C | L | 76 | 4C | 1 | 108 | *6C* |
| CtrlM | CR | 13 | 0D | - | 45 | 2D | M | 77 | 4D | m | 109 | 6D |
| CtrlN | SO | 14 | 0E | . | *46* | 2E | N | 78 | 4E | n | 110 | 6E |
| CtrlO | SI | 15 | OF | / | 47 | 2F | 0 | 79 | 4F | **o** | 111 | 6F |
| CtrlP | DLE | 16 | 10 | 0 | 48 | 30 | P | 80 | 50 | p | 112 | 70 |
| CtrlQ | DC1 | 17 | 11 | 1 | 49 | 31 | Q | 81 | 51 | q | 113 | 71 |
| CtrlR | DC2 | 18 | 12 | 2 | 50 | 32 | R | 82 | 52 | r | 114 | 72 |
| CtrlS | DC3 | 19 | 13 | 3 | 51 | 33 | S | 83 | 53 | s | 115 | 73 |
| CtrlT | DC4 | 20 | 14 | 4 | 52 | 34 | T | 84 | 54 | t | 116 | 74 |
| CtrlU | NAK | 21 | 15 | *5* | 53 | 35 | U | 85 | 55 | u | 117 | 75 |
| CtrlV | SYN | 22 | 16 | 6 | 54 | 36 | V | 86 | 56 | **V** | 118 | 76 |
| Ctrl W | ETB | 23 | 17 | 7 | 55 | 37 | w | 87 | 57 | w | 119 | 77 |
| CtrlX | CAN | 24 | 18 | 8 | 56 | 38 | X | 88 | 58 | **X** | 120 | 78 |
| CtrlY | EM | 25 | 19 | 9 | 57 | 39 | Y | 89 | 59 | y | 121 | 79 |
| CtrlZ | SUB | 26 | 1A | : | 58 | 3A | Z | 90 | 5A | *z* | 122 | 7A |
| Ctrl[ | Esc | 27 | IB | ; | 59 | 3B | [ | 91 | 5B | *{* | 123 | 7B |
| Ctrl\ | FS | 28 | 1C | < | 60 | 3C | \ | 92 | 5C |  | 124 | 7C |
| Ctrl] | GS | 29 | ID | = | 61 | 3D | ] | 93 | 5D | *}* | 125 | 7D |
| Ctrl^ | RS | 30 | IE | > | 62 | 3E | **A** | 94 | 5E |  | 126 | 7E |
| Ctrl | US | 31 | IF | ? | 63 | 3F |  | 95 | 5F |  | 127 | 7F |

设明文为，密钥矩阵（或线性变换）为可逆的矩阵**K**，则经过线性变换得密文。其中



或写成矩阵形式：



其中



解密得明文**M**：



于是，我们设置如下实验步骤：

1. 将明文对照ASCII编码表进行数字编码，并创建为5\*5的明文矩阵，若明文的字符长度不够则最后添加空格代替；

2. 通过randi()函数创建随机密钥矩阵**K**（随机矩阵n阶始终为逆矩阵，可以作为密钥矩阵）；

3. 代入密钥矩阵**K**和明文矩阵**M**，通过公式将明文转化为密文；

4. 通过求解**K-1**并将其代入公式，可以解密出明文矩阵**M**。

5.对照设置好的字符编码表将明文矩阵**M**转化为字符语句，完成解密过程。

#### 实验程序

%问题一代码

clc

%将明文数字编码为矩阵M

M = [77 108 100 106 100;

114 108 101 115 101;

32 32 32 32 32;

72 109 116 99 32;

106 97 104 111 32]

%%第一组密钥矩阵K1

disp('第一组密钥矩阵：')

K1 = randi([0 50],size(M))

disp('第一次加密后密文为：')

C = K1\*M %加密密文

disp('第一组密钥矩阵逆矩阵：')

inv(K1)

disp('第一次解密后明文为：')

M\_ = K1\C %解密密文

%%第二组密钥矩阵K1

disp('第二组密钥矩阵：')

K2 = randi([0 50],size(M))

disp('第二次加密后密文为：')

C = K2\*M %加密密文

disp('第二组密钥矩阵逆矩阵：')

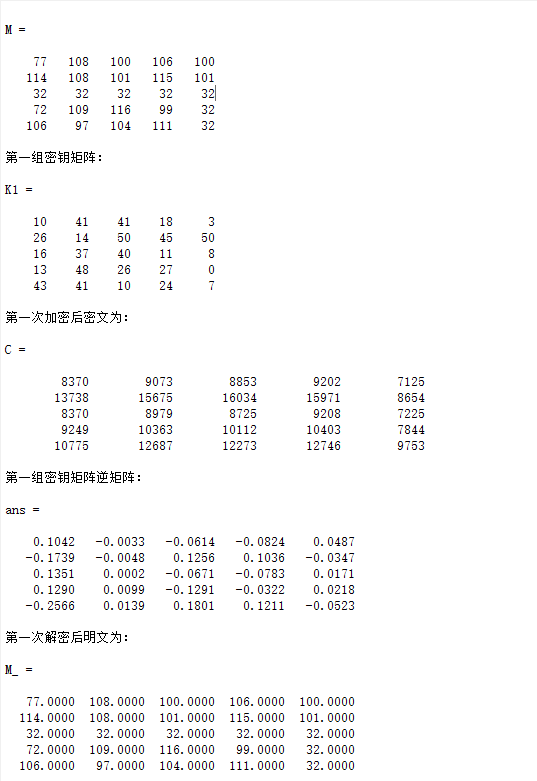
inv(K2)

disp('第二次解密后明文为：')

M\_ = K2\C %解密密文

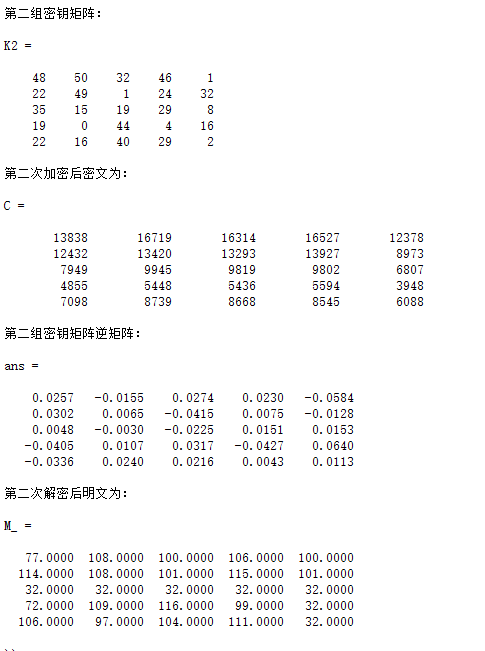
#### 实验结果

1. 运行第一节结果：



**Figure 1 第一节运行结果**

1. 运行第二节结果：



**Figure 2 第二节运行结果**

#### 结果分析

通过建立randi()函数建立两组随机矩阵，取值范围设置在[0, 50]之间，由于MATLAB的随机矩阵默认为可逆矩阵，因此满足题意。

第一组取的随机矩阵为:



其逆矩阵为:



第一次加密后的密文矩阵为：



第二组取的随机矩阵为:



其逆矩阵为:



第二次加密后的密文矩阵为：



实验结果证明，两次解密得到的结果都与原明文矩阵相同，实验结果正确。

### 问题2

甲方截获了一段密文：OJWPISWAZUXAUUISEABAUCRSIPLBHAAMMLPJJOTENH（中间没有空格）。经分析这段密文是用Hill2（密钥矩阵为2阶矩阵）密码编译的，且这段密码的字母UCRS一次代表字母TACO，问能否破译这段密文的内容？

若已知密文的前四个字母OJWP分别代表TACO，问能否将此密文破译？（提示：可用程序穷举的方法找到满足方程的 2 阶密钥矩阵。）

#### 问题分析

当四个字母UCRS代表TACO：

首先，分析密文由26个英文字母（全大写）组成，因此我们定义一个数字编码格式如Tab-2所示：

**Table 2 26个英文字母、空格和句号对应数字编码**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **…** | **X** | **Y** | **Z** |
| 1 | 2 | 3 | 4 | … | 24 | 25 | 26 |

将截获到的密文按照十进制数字编码转化为明文代码：

15 10 23 16 9 19 23 1 26 21 24 1 21 21 9 19 5 1 2 1 21 3 18 19 9 16 12 2 8 1 1 13 13 12 16 10 10 15 20 5 14 8。然后得到密文矩阵：



接着需要计算密钥矩阵K，题目已知条件UCRS(21 3 18 19)分别代表TACO(20 1 3 15)，利用模m逆矩阵计算公式可得：





接着通过还原明文矩阵。甲方收到密文后可以用解密矩阵**K-1** 采用对应的算法进行解密得明文矩阵M：



再将明文矩阵**M**转换成明文代码。对照字母表，甲方可以得到原文信息，根据语言习惯将其标准化可以得到原文。

同理，当四个字母OJWP代表TACO，也如上面一样进行分析。

于是，我们设置如下实验步骤：

1. 通过已知信息得到2\*2的密文矩阵和2\*2原文矩阵；

2. 通过公式计算出当前密文矩阵，若**K-1**不存在，则利用try-catch捕捉错误返回全0值；

3. 将**K-1**其代入公式，可以解密反推出完整的明文矩阵**M**。

4. 将明文矩阵对照字母表转化为字符语句，解密完成。

#### 实验程序

%问题二代码

clc

clear all

%将密文数字编码为密文矩阵C

C = [15 23 9 23 26 24 21 9 5 2 21 18 9 12 8 1 13 13 10 20 14;

10 16 19 21 21 1 21 19 1 1 3 19 16 2 1 13 12 10 15 5 8];

m = 26; %模为26

%%若四个字母UCRS代表TACO

result1 = [21 18; %若四个字母UCRS代表TACO

3 19];

result2 = [15 23;

10 16];

test1 = [20 3; %测试字母TACO

1 15];

mod\_det\_result1 = mod(det(result1),m); %求det(C)(mod26)

for i=1:m %求inv\_det(C)(mod26)

if round(mod(i\*m+1,mod\_det\_result1)) == 0 %需要round函数转化为整数

imod\_det\_result1 = (i\*m+1)/mod\_det\_result1;

end

end

try %用try catch检查是否存在imod\_det\_result1，不存在即赋值0

imod\_det\_result1

catch

imod\_det\_result1 =0

end

inv\_K = test1\*imod\_det\_result1\*det(result1)\*mod(inv(result1),m) %inv\_K = M\*inv\_C(mod26)

disp('解密后明文为：')

M\_ = mod(inv\_K\*C,26) %解密密文

%%若四个字母OJWP代表TACO

mod\_det\_result2 = mod(det(result2),m); %求det(C)(mod26)

for i=1:m %求inv\_det(C)(mod26)

if round(mod(i\*m+1,mod\_det\_result2)) == 0 %需要round函数转化为整数

imod\_det\_result2 = (i\*m+1)/mod\_det\_result2;

end

end

try %用try catch检查是否存在imod\_det\_result2，不存在即赋值0

imod\_det\_result2

catch

imod\_det\_result2 =0

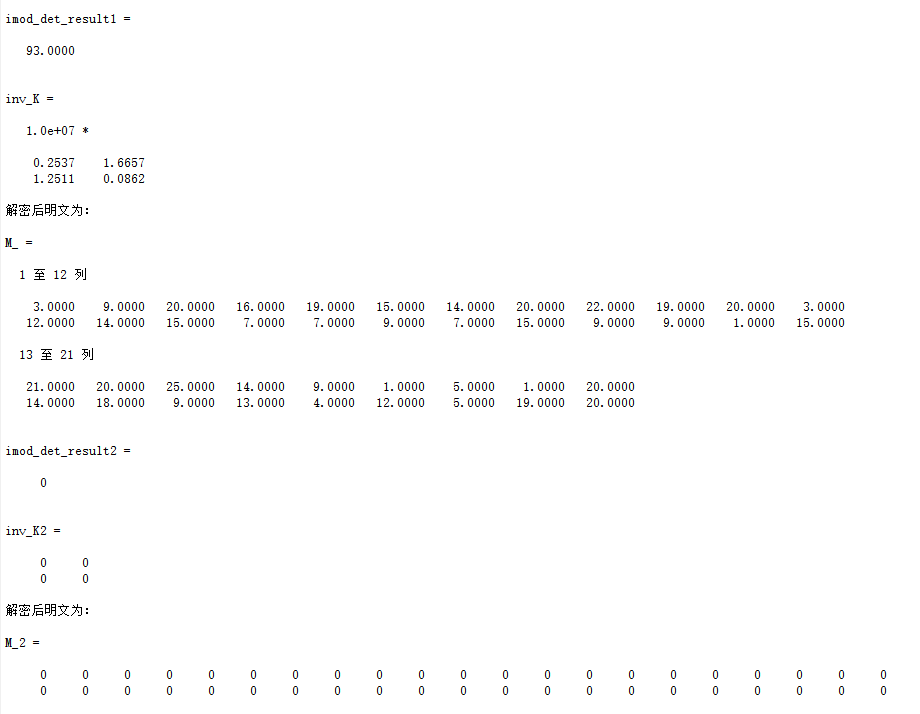
end

inv\_K2 = test1\*imod\_det\_result2\*det(result2)\*mod(inv(result2),m) %inv\_K2 = M\*inv\_C(mod26)

disp('解密后明文为：')

M\_2 = mod(inv\_K2\*C,26) %解密密文

#### 实验结果



#### 结果分析

若四个字母UCRS代表TACO，则可以根据公式计算出密钥矩阵的逆矩阵inv\_K，再根据公式代入密文矩阵进行解密，得到解密后的明文矩阵，提取其中的明文代码：3 12 9 14 20 14 9 19 7 15 9 14 7 20 15 22 9 19 9 20 1 3 15 21 14 20 18 25 9 14 13 9 4 4 12 5 5 1 19 20 20，对照字母-数字转换表将其转化为大写英文字母：CLINTONISGOINGTOVISITACOUNTRYINMIDDLEEASTT，根据语言习惯将其标准化（同时去掉最后一个字母t）可以得到原文：Clinton is going to visit a country in Middle East.

若四个字母OJWP代表TACO，发现try-catch捕捉到错误，即不存在K-1，因此无法解密。

## 实验感想

通过本次实验操作，我基本理解了加密解密的基本原理、Hill密码加密解密原理以及相关算法、模型及应用的掌握，同时复习了一些矩阵运算知识和数学知识（如辗转相除法）。

我认为这次实验是非常有意义有价值的，通过这次案例学习，我对代数运算的相关知识和应用有了更深的理解，初步掌握了一种加密解密的方法。在运行程序的过程中出了一些报错，我经过仔细核对和调试，最终都解决了它们，通过这些解决经验我对常见的报错（比如矩阵维数不对、要用round()函数取整等）有了更进一步的认识，这有助于我在后面的实验中降低出现类似错误出现的概率，同时我也进一步学会了try-catch的使用。在这次实验中涉及到的疑难知识点、用到的新函数我都通过记笔记或者录屏的方式认真记了下来，丰富了我的MATLAB知识储备。在本次实验中，所有的实验均由我独立完成，相关代码和图片结果也都整理到位，代码中存在疑惑的地方以及需要注意的地方均已注释好，以备下次复习时使用。

在这次实验里，我认真完成了两个实验任务，颇有所获，相信未来几次实验会继续收获不少新知识。

6 许柏城 62 课外实验4

2020-04-12 19:00