CRYPTANALYSIS PROJECT: HILLY CIPHER

Vishrut Sharma

INDEX

Problem Introduction	2
Encryption Procedure	3
Decryption Procedure	4
Attack 1	5
Attack 2	14
Conclusion	15
References	15

Problem Introduction:

The problem described is a variation of the hill cipher: A polygraphic substitution cipher based on linear algebra.

The author modifies the hill cipher in the following ways and calls it the “Hilly Cipher”

- Modify the substitution table for plaintext characters(A-Z) , so that it is an addicted to the key.
- The conventional hill cipher provides inverse modulo 26, which is used while solving linear equations during decryption. To circumvent this, the author chooses a key, for which determinant of inverse mod 26 does not exist.

The author provides a plain text and cipher text pair (P1, C1)

Cipher text: C1

```
L?%$^-@-?%J%-?&*-#^&P%^*$$-#$%P?+-&^%#%*Z%*%^*---+X?%%$?^*@-T%&?-
?@@&?F%?+&$+&##B%$*%$#-#-H?%%*?#+&^X%-##$??@-N%^@-?^*^*P%#%%**#?-Z$+%@*&^%Z*-
#^?^??J%?&$@+-#@Z***+-&?@F&-?&$#$$V%+%$*?-%^N&--%@@-%Z%#+$$^-^*F%-*@#^#*-
T%##@@#$@@X%?$^@?*%%Z%@$-&#%$$Z&*$^*^$#D%?-??#-&@T%$@&%^++^Z%%&*&$-%@V%@+-
^*&^*P%##&$^?--F$^$%*%&^D*^^^-*@?D%?-+#%&#$T$&?@@*%+N&@@*+&+$
```

Corresponding plain text : P1

```
HILLYISANIMPROVEDVERSIONOFHILLCIPHER
```

The author also provides another Cipher text : C2

```
L%-+^---#+D&$@%#*@^B%++**@$$-H%$&#%^@%%V%+*?$-$-^R%--#@?@%^J%-^?#?-%-
R%%##??##+X%?+&$&^&-N%&-%?*&%@L%?@$*@+#*N%-+$&$&$^P%^$$$??*?X%++$@?&%+V%-
?+?$??+N?%&-***^$L%+$-*-*@?D%**^*-+-*B%^?&-$*@^X%%*#^#@&+R%-**&%^%*H?#+?%+^?&V%@++-
^@^@Z%*$&?%%+#D%&?-?+%$%R%?^*$%+?-Z%$-@^#^$*Z?#*$&$+%-J%--*$@$$&T%^@*-
*@%$T**+@?%#$Z-^##*@#&J-@?#?^$^L%%$+$$?*F$^@?-@+&L^?#@#%@#D%?-^#-
*?#R^$$&*?+&P%#?^*@%#+V%@^++*#$?J&@#^$&^^J%?##@*^#*P%+++*-
%@#F%#$@??^%@X%#$$+%%$$Z%$#^&#@**L%%-$*^^&^H%+-^*#&@?D%%^%^@$$$T^-#@-&-%J^^$-
%?^#J%-?-%@@?-T%#%+^%#&%B%?%&#&?&%H&$&%%$@#D$#?%%#%$H**+###&#P%%^??---*F$%*-%-
$?B^**+%$#&H%#$?%@@$+F*#$*-*^-H^$^+#*%+X%?+@^-%?&R$^&+*&-&L&-*+%#**
```

Other hints about the problem :

- Both cipher texts C1 and C2 are encrypted with the same keys.
- Cipher text C1 is not part of cipher text C2.

The author also provides an additional lookup table with 11 characters (*, #, &, %, $, +, ?, !, @, ^, -).

This table is rotated and is used as a substitution table for numbers during encryption.

*	#	&	%	$	+	?	!	@	^	-
-	0	1	2	3	4	5	6	7	8	9

Encryption procedure :

- Key is chosen as an invertible n*n matrix. K with n >= 5 , or n = 6 by default.
- K must follow the condition gcd (det(K), 26) != 1.
- Compute s = sum of elements of main diagonal of key (K)
  →If s = 0 , then s = sum of elements of secondary diagonal.
  →If this is also 0 then : Generate new key matrix.
- Compute substitution table Tp as
  A = 1* s
  B = 2*s
  C= 3*s
  .
  .
  .
  Z = 26 * s
- Divide plain text into blocks of n(6) characters .
- Replace plain text with numbers using substitution table in Tp. This results in a plaintext number vector . P
- Multiply K and P to get vector : C1 = K * P
- Compute vector for cipher text C = C1 mod 26
- Substitute elements of C like A = 1 ; B = 2 ; C= 3 and so onn.
- Compute an extension vector X as
  X = upper bound(C1 / 26)
- Construct an extension matrix E as
  E = s* I + k
- Compute the encrypted extension as Xenc = E * X
- Append all the cipher letters with the encrypted extensions.
- C1 XEnc1 ... C6 XEnc6
- Define the additional symbols used for substituting digits as the following table

*	#	&	%	$	+	?	!	@	^	-
-	0	1	2	3	4	5	6	7	8	9

- Compute sum "t" of all elements of key matrix K .
- Upon this result rotate the above substitution table .

Eg: t mod 11 = 6
Shift table 6 times to right

*	#	&	%	$	+	?	!	@	^	-
+	?	!	@	^	-	*	#	&	%	$

- Substitute XEnc using the substitution table TA. So you get XEncSym.
- Send C1 XEncSym …….C6 XEncSym as cipher text to recipient.

Decryption Procedure

Given a key matrix K and the corresponding cipher text

- Compute sum of diagonal of key matrix to get s and table Tp for plaintext character alphabets.
- Calculate sum of all elements of key and perform mod11. Implement the substitution table Ta for additional symbols. Now substitute the cyphertext characters using the substitution table Ta to get the extensions Xenc
- Substitute all cyphertext letters from C using the substitution table TC: A=1, B=2,..., Z=26.
- Compute matrix E as s*I + k (I is the matrix of size n*n containing 1)
- Compute the inverse K-1 of K and E -1 of E .
- Multiply E-1 and Xenc to get the vector "extension vector " X = E-1 * Xenc
- Using the formula C1 = (X * 26) - (26 - C) find C1.
- Perform multiplication of K -1 by C1 (this must be done with exact fractions only) .
- P = K -1 * C1
- Substitute P with the corresponding letters from table Tp.
- (Note : The decryption required exact fractions, hence we performed P = k-1* C1 in MATLAB and not C)

Attack 1 :

Both the cipher text C1 and C2 are encrypted with the same key. The length of C1 is 36 and that of C2 is 66 , here by length we mean the actual letters and not the symbols. Also, both cipher text doesn't involve any padding and this assumption is validated.

The problem also specifies the following procedure for padding the cipher text :

- R is calculated as = n – (letters in plaintext mod n).
  Example: R = 6 - (8 mod 6)
  6-2 = 4
- If R ==n , it implies that no padding is needed.
- Substitute R using the following substitution table Ts for symbols:

- Append letters to the end of the cyphertext equal to the value of R
  (Example: The symbol in Ts for 4 is, ". EXAMPLE = **EXAMPLEEEEEE,**)

Both the cipher texts C1 and C2 doesn't have symbols matching the above 6 values and hence our assumption that no padding is involved is validated.

Then the size of the key matrix K is assumed to be 6. This is done by taking the gcd of 36 and 66.

Cyphertext	**L**	**J**	**P**	**P**	**Z**	**X**
	12	10	16	16	26	24
Plain text	H	I	L	L	Y	I
	8	9	12	12	25	9
Cyphertext	**T**	**F**	**B**	**H**	**X**	**N**
	20	6	2	8	24	14
Plain text	S	A	N	I	M	P
	19	1	14	9	13	16
Cyphertext	**P**	**Z**	**Z**	**J**	**Z**	**F**
	16	26	26	10	26	6
Plain text	R	O	V	E	D	V
	18	15	22	5	4	22
Cyphertext	**V**	**N**	**Z**	**F**	**T**	**X**
	22	14	26	6	20	24
Plain text	E	R	S	I	O	N
	5	18	19	9	15	14
Cyphertext	**Z**	**Z**	**D**	**T**	**Z**	**V**
	26	**26**	**4**	**20**	**26**	**22**
Plain text	O	F	H	I	L	L
	15	6	8	9	12	12
Cyphertext	**P**	**F**	**D**	**D**	**T**	**N**
	16	6	4	4	20	14

Plain text	C	I	P	H	E	R
	3	9	16	8	5	18

Plain text : HILLY IS AN IMPROVED VERSION OF HILL CIPHER

Each block of plaintext is **encrypted** as :
Multiply K by P to get the vector C1: C1 = K * P
Then compute the cyphertext vector C = C1 mod 26.

a	**b**	**c**	**d**	**e**	**f**	*	s*8	**mod 26 =**	**12**
g	h	i	j	k	l	*	s*9	mod 26 =	10
m	n	o	p	q	r	*	s*12	mod 26 =	16
s	t	u	v	w	x	*	s*12	mod 26 =	16
y	z	a1	b1	c1	d1	*	s*25	mod 26 =	26
e1	f1	g1	h1	i1	j1	*	s*9	mod 26 =	24

Where each element in P is multiplied by s = sum of the leading diagonal elements
That can be rewritten as

a *s	**b*s**	**c*s**	**d*s**	**e*s**	**f*s**	*	**8**	**Mod26 =**	**12**
g*s	h*s	i*s	j*s	k*s	l*s	*	9	Mod 26 =	10
m*s	n*s	o*s	p*s	q*s	r*s	*	12	Mod 26=	16
s*s	t*s	u*s	v*s	w*s	x*s	*	12	Mod26 =	16
y*s	z*s	a1*s	b1*s	c1*s	d1*s	*	25	Mod26 =	26
e1*s	f1*s	g1*s	h1*s	i1*s	j1*S	*	9	Mod26 =	24

Considering the element of the key matrix say a* s as the unknown and call it x00
(The same is applied to all the elements) and we get the below equation.

x00	**x01**	**x02**	**x03**	**x04**	**x05**	*	**8**	**Mod26=**	**12**
x10	x11	x12	x13	x14	x15	*	9	Mod26 =	10
x20	x21	x22	x23	x24	x25	*	12	Mod26 =	16
x30	x31	x32	x33	x34	x35	*	12	Mod26 =	16
x40	x41	x42	x43	x44	x45	*	25	Mod26 =	26
x50	x51	x52	x53	x54	x55	*	9	Mod26 =	24

We have a system of linear equations of the form:

(8*x00+ 9 *x01+12*x02+12 * x03+ 25 * x04+ 9 * x05) mod 26 = 12
(8*x10+ 9 *x11+12*x12+12 * x13+ 25 * x14+ 9 * x15) mod 26 = 9
(8*x20+ 9 *x21+12*x22+12 * x23+ 25 * x24+ 9 * x25) mod 26 = 16

(8*x30+ 9 *x31+12*x32+12 * x33+ 25 * x34+ 9 * x35) mod 26 = 16
(8*x40+ 9 *x41+12*x42+12 * x43+ 25 * x44+ 9 * x45) mod 26 = 26
(8*x50+ 9 *x51+12*x52+12 * x53+ 25 * x54+ 9 * x55) mod 26 = 24

For each such block of 6 plain text we get 6 equations so 36 equations with 36 unknowns.

So, we consider 6 equations and solve them

8	**9**	**12**	**12**	**25**	**9**	*****	**x00**	**Mod26**	**12**
19	1	14	9	13	16	*	x01	Mod26	20
18	15	22	5	4	22	*	x02	Mod26	16
5	18	19	9	15	14	*	x03	Mod26	22
15	6	8	9	12	12	*	x04	Mod26	26
3	9	16	8	5	18	*	x05	Mod26	16

We solved the above equations using MATLAB code as below:

```
A = [8 9 12 12 25 9;
19 1 14 9 13 16;
18 15 22 5 4 22;
5 18 19 9 15 14;
15 6 8 9 12 12;
b = [12;20;16;22;26;16];
s = linsolve(sym(A),sym(b))

x00=1092354/480385 =2.27
x01=64752/96077 =0.67
x02=1864486/480385 =3.88
x03=1218732/96077 =12.68
x04=-2793018/480385 = - 5.81
x05=-3506706/480385 = -7.299
```

Similarly solving for the rest we get :

X10=	378994/480385	0.7889
X11=	241386/96077	2.512
X12=	-1879454/480385	-3.91
X13=	-177872/96077	-1.85
X14=	448922/480385	0.9345
X15=	1434694/480385	2.98655
X20=	-2258/96077	-0.0235
X21=	132368/96077	1.377
X22=	-25292/96077	-0.263

X23=	-415726/96077	-4.327
X24=	186270/96077	1.938
X25=	111050/96077	1.1558

X30=	-59148/480385	-0.123
X31=	116738/96077	1.215
X32=	-1993472/480385	-4.1497
X33=	-348140/96077	-3.6236
X34=	1172566/480385	2.44
X35=	2044672/480385	4.2563

X40=	-69654/480385	-0.145
X41=	54534/96077	0.5676
X42=	-1189526/480385	-2.47619
X43=	-231312/96077	-2.40757
X44=	929888/480385	1.9357
X45=	1722116/480385	3.5848

X50=	484166/480385	1.007
X51=	67014/96077	0.6975
X52=	919724/480385	1.91455
X53=	788720/96077	8.209
X54=	-1409602/480385	-2.9343
X55=	-2053284/480385	-4.2742

→

Mod 26 of the fractional values were found using the method
480385^-1 mod 26 =3 and 96077^-1 mod 26 =15
1092354/480385 mod 26 = (1092354 *(480385^-1 mod 26))mod 26=22
By doing so the values for the 36 unknowns are:

KS= [22 24 26 16 18 2

2 4 24 14 18 16

8 4 12 2 12 8

6 22 26 26 2 18

26 24 26 20 20 18

8 24 26 20 16 16]

So the above matrix obtained is **(K*s) mod 26** , let's call it KS . Here we have substituted the value 26 instead of 0.

We know that

(A * B) mod C = (A mod C * B mod C) mod C

So in our case it is (K * P) mod 26 = (K mod 26 * P mod 26) mod 26 = C

As our 's' value is included in the K matrix, we have the values A=1, B=2,C=3… in our P matrix and so it's the same value on taking mod 26 except for Z=26 which gives a mod value of 0. This we can resubstitute as 26 instead of 0.

So applying the similar logic we can write it as below:

(KS * P) mod 26 = C

We know the values of KS and P so solving for it we get C as:

22	24	26	16	18	2		8	1364		12
2	4	24	14	18	16		9	1102		10
8	4	12	2	12	8	*	12	= 640	mod 26 =	16
6	22	26	26	2	18		12	1082		16
26	24	26	20	20	18		25	1638		26
8	24	26	20	16	16		9	1376		24

Which matches the first 6 cipher text letters L,J,P,P,Z,X
Repeating the above method for other plain text we get
The given cipher text letters as

LJPPZXTFBHXNPZZJZFVNZFTXZZDTZVPFDDTN which matches what is given for C1.

We can use the KS matrix and use the cipher text 2 C2 and use it to solve for the 66 plain text values. Here we are trying to solve with the letters directly than considering the whole cipher text with symbols.

The cipher text to be solved for along with it's alphabet letters are given by :

CipherText	L	D	B	H	V	R
	12	4	2	8	22	18
CipherText	**J**	**R**	**X**	**N**	**L**	**N**
	10	18	24	14	12	14
CipherText	**P**	**X**	**V**	**N**	**L**	**D**

	16	24	22	14	12	4
CipherText	**B**	**X**	**R**	**H**	**V**	**Z**
	2	24	18	8	22	26
CipherText	**D**	**R**	**Z**	**Z**	**J**	**T**
	24	**18**	**26**	**26**	**10**	**20**
CipherText	**T**	**Z**	**J**	**L**	**F**	**L**
	20	26	10	12	6	12
CipherText	**D**	**R**	**P**	**V**	**J**	**J**
	4	18	16	22	10	10
CipherText	**P**	**F**	**X**	**Z**	**L**	**H**
	16	6	24	26	12	8
CipherText	**D**	**T**	**J**	**J**	**T**	**B**
	4	20	10	10	20	2
CipherText	**H**	**D**	**H**	**P**	**F**	**B**
	8	4	8	16	16	2
CipherText	**H**	**F**	**H**	**X**	**R**	**L**
	8	6	8	24	18	12

(KS * X) mod 26 = C
Substituting the values for KS and C for the first 6 letters we get

22	**24**	**26**	**16**	**18**	**2**		**x00**		**12**
2	4	24	14	18	16		x01		4
8	4	12	2	12	8	*	x02	mod 26=	2
6	22	26	26	2	18		x03		8
26	24	26	20	20	18		x04		22
8	24	26	20	16	16		x05		18

Which is is linear equation with 6 unknowns

(22*x00+24 * x01+26 *x02+16 * x03+18 *04+2*x05)mod 26 = 12
(2*x00+4 * x01+24 *x02+14 * x03+18 *04+16*x05)mod 26 = 4
(8*x00+4 * x01+12*x02+2* x03+12 *04+8*x05)mod 26 = 2
(6*x00+22 * x01+26 *x02+26 * x03+2 *04+18*x05)mod 26 = 8
(26*x00+24 * x01+26 *x02+20 * x03+20 *04+18*x05)mod 26 = 22
(8*x00+24 * x01+26 *x02+20 * x03+16 *04+16*x05)mod 26 = 18

The above 6 linear equations were solved using a Matlab code as below :

```
A = [   22 24 26 16 18 2;
        2 4 24 14 18 16;
        8 4 12 2 12 8;
```

```
        6 22 26 26 2 18;
        26 24 26 20 20 18;
        8 24 26 20 16 16];
b = [12;4;2;8;22;18];
s = linsolve(sym(A),sym(b))
```

x00	**-5008/132211**	**mod 26=**	**10**	**J**
x01	92747/132211	mod 26=	5	E
x02	-185351/132211	mod 26=	3	C
x03	111177/132211	mod 26=	1	A
x04	135149/132211	mod 26=	1	A
x05	39196/132211	mod 26=	14	N

Here the (inverse of 132211) mod 26 = 1 so

-5008/132211 mod 26= is = -5008 mod 26 =10

X10	**10220/132211**	**mod 26=**	**2**	**B**
X11	53502/132211	mod 26=	20	T
X12	404969/132211	mod 26=	19	S
X13	-475148/132211	mod 26=	2	B
X14	-190373/132211	mod 26=	25	Y
X15	156555/132211	mod 26=	9	I
X20	101327/132211	mod 26=	5	E
X21	-137772/132211	mod 26=	2	B
X22	455840/132211	mod 26=	8	H
X23	-306067/132211	mod 26=	5	E
X24	-213990/132211	mod 26=	16	P
X25	44881/132211	mod 26=	5	E
x30	-81573/132211	mod 26=	15	O
x31	99556/132211	mod 26=	2	B
x32	-24259/132211	mod 26=	25	Y
x33	-154487/132211	mod 26=	5	E
x34	130910/132211	mod 26=	0	Z
x35	207915/132211	mod 26=	19	S

x40	-0.081710296	mod 26=	13	M
x41	106006/132211	mod 26=	4	D
x42	629320/132211	mod 26=	16	P
x43	-615771/132211	mod 26=	13	M
x44	-322553/132211	mod 26=	3	C
x45	81278/132211	mod 26=	2	B
x50	-32599/132211	mod 26=	5	E
x51	-104189/132211	mod 26=	19	D
x52	237108/132211	mod 26=	14	N
x53	-7189/132211	mod 26=	13	M
x54	1331/132211	mod 26=	5	B
x55	-105904/132211	mod 26=	20	T
x60	32170/132211	mod 26=	8	H
x61	-32881/132211	mod 26=	9	I
x62	334387/132211	mod 26=	1	A
x63	-281686/132211	mod 26=	24	X
x64	-214127/132211	mod 26=	9	I
x65	138724/132211	mod 26=	14	N
x70	111677/132211	mod 26=	7	G
x71	65439/132211	mod 26=	23	W
x72	619520/132211	mod 26=	18	R
x73	-641594/132211	mod 26=	8	H
x74	-448052/132211	mod 26=	6	F
x75	155433/132211	mod 26=	5	E
x80	126871/132211	mod 26=	17	Q
x81	-189920/132211	mod 26=	10	J
x82	36265/132211	mod 26=	21	U
x83	92253/132211	mod 26=	5	E
x84	-15664/132211	mod 26=	14	N
x85	79388/132211	mod 26=	10	J
x90	73091/132211	mod 26=	5	E
x91	-24892/132211	mod 26=	16	P
x92	270275/132211	mod 26=	5	E
x93	-196776/132211	mod 26=	18	R
x94	-217489/132211	mod 26=	1	A
x95	41581/132211	mod 26=	7	G
x10 0	67384/132211	mod 26=	18	R
x10 1	26654/132211	mod 26=	4	D
x10 2	140550/132211	mod 26=	20	T
x10 3	-125442/132211	mod 26=	8	H
x10 4	-163717/132211	mod 26=	5	E

x10 5	117611/132211	mod 26=	13	M

The plain text obtained by this method is

JECAANBTSBYIEBHEPEOBYEZSMDPMCBESNMBTHIAXINGWRHFEQJUENJEPERAGRDTHEM

Thus ONLY FEW dictionary words and when validated against the result in the competition it failed.

But "BHEPEOBYE" maybe we can consider this substring is "THE PEOPLE" but the other details cannot be obtained as the cipher text provides both confusion and diffusion. So frequency analysis and digraph statistics also can't be used on cipher text but can be used after decryption. It can be considered as a score or the number of dictionary words can be used as a scoring mechanism.

But in this case we known a plain text and cipher text pair and both the cipher texts are encrypted with the same key matrix. So we can try an exhaustive search for the key matrix of size 6*6 and each element can be of any range positive or negative.

One other option before trying exhaustive search was to try to determine the value of s , from the matrix KS the common multiple was 2 which was removed and then multiplied with the plaintext and when we used this method to solve the problem even those results matched the one from the above attack. Moreover the sum of diagonal elements of the Key matrix K thus obtained wasn't equal to s, hence this method was discarded.

Attack 2:

As a last resort we tried to attack the cipher to obtain the key by using Exhaustive search. Here we tried to substitute different values for the 36 unknown and generate the key matrix and encrypt the plain text P1 with the key and check the cipher Text if it was matching.

To reduce the search space we made use of the fact that the GCD of the determinant of the key matrix and 26 must be one of the numbers (2, 4, 6, 8, 10, 12, 13, 14, 16, 18, 20, 22, 24) this is due the fact that the key matrix is invertible but not invertible modulo 26. If the GCD of the determinant of the matrix and 26 if not in the list or if the determinant of the matrix is 0 the forthcoming steps to encrypt the plain text P1 and to check if it matches the given actual cipher text C1 is skipped.

The key matrix is generated as follows

```
[0 0 0 0 0 0   [0 0 0 0 0 0   [0 0 0 0 0 0        [0 0 0 0 0 0        [0 0 0 0 0 0
 0 0 0 0 0 0    0 0 0 0 0 0    0 0 0 0 0 0         0 0 0 0 0 0         0 0 0 0 0 0
 0 0 0 0 0 0    0 0 0 0 0 0    0 0 0 0 0 0   ....    0 0 0 0 0 0   ....    0 0 0 0 0 0
 0 0 0 0 0 0    0 0 0 0 0 0    0 0 0 0 0 0         0 0 0 0 0 0         0 0 0 0 0 0
```

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0] 0 0 0 0 0 1] 0 0 0 0 0 2] 0 0 0 0 1 0] 0 0 0 1 0 0]

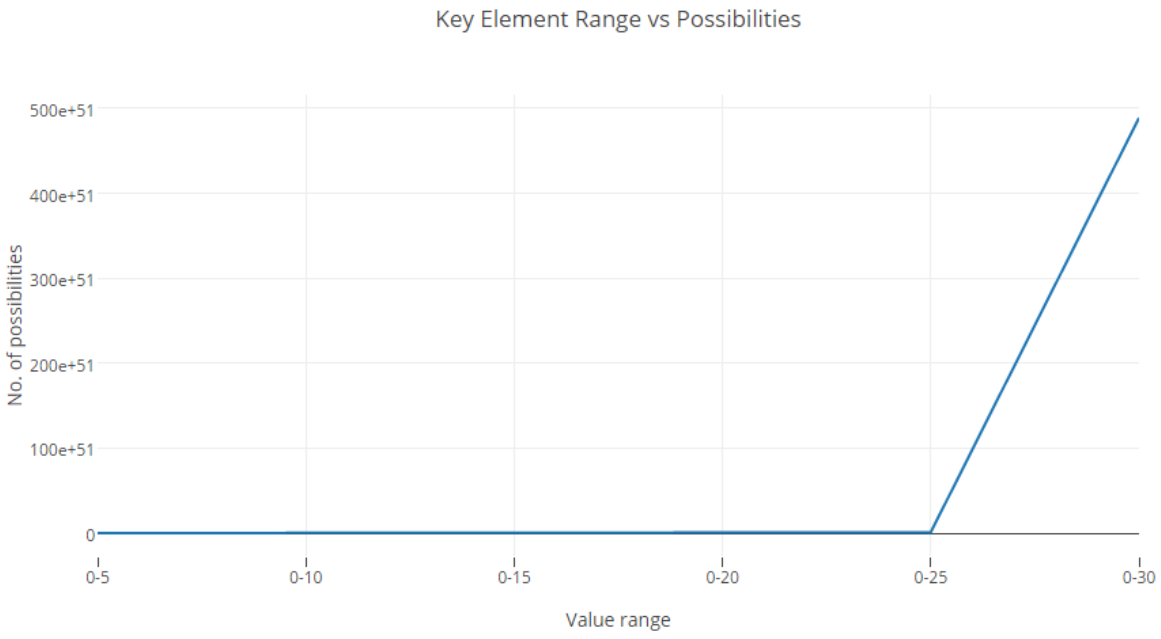
The values here go from 0 to 20 for each element in the matrix. 20 possible values for each unknown and there are a total of 36 unknowns that's around (21)^36.

We tested it by running a C program for the attack where we generated the key matrix , verified if determinant is greater than 0 and then checked the GCD of the determinant with 26 if it's not one of (2, 4, 6, 8, 10, 12, 13, 14, 16, 18, 20, 22, 24) then we generate the next matrix and so on ..

If it has determinant in the list then it proceeds with the encryption and verification if it matches the cipher Text C1. If a match is found then we stop and use that key to decrypt the plain text P2.

The program generates and searches 5632 different possibilities each second and in a minute it can verify 332394 possibilities.

The graph below shows the range of values and no of different possible values that has to be searched in:



In one day 486604800 possibilities are calculated

In one year 177610752000 possibilities are calculated

Range	Possibilities	Years taken to compute
0-5	1.03E+28	5.81E+16
0-10	3.09E+37	1.74E+26
0-15	2.23E+43	1.26E+32
0-20	3.98E+47	2.24E+36
0-25	8.69E+50	4.89E+39
0-30	4.89E+53	2.75E+42

Conclusion :

Exhaustive search will take many years to crack the hilly cipher. Hence a more intelligent approach is needed. One way to solve the cipher is using Linear Algebra, which has been explored by us in the report.

More robust techniques involving modulo Linear Algebra need to be explored to crack the cipher.

References :

1. Code for calculating inverse of a matrix – Code champ
   http://www.ccodechamp.com/c-program-to-find-inverse-of-matrix/
2. Shifting lookup the lookup table - tech crash course
   http://www.techcrashcourse.com/2016/07/program-for-array-rotation-N-positions.html

3.Mystery Twister : forums

https://www.mysterytwisterc3.org/en/