Cryptography Systems Series #1

Antiquated Ciphers Part 1

Polygraphic Ciphers based on Linear Algebra

By Macarthur Inbody

Version 0.6

Editing Time: 08:26:28 (HH:MM:SS)

Table of Contents

Introduction

The topics covered this time will be all ciphers that involve pure mathematics that are more complex than simple shift ciphers. So it’s going to be Hill and also Affine ciphers.

For the hill cipher we’re going to be writing a solver and going over the Hill cipher by hand. It will only be a 2x2 matrix. The order will go like this. First we will go over some background of the cipher. After that we’ll encrypt a message and then we’ll decrypt the same message. Finally a solver will be written that is able to solve the message with a known crib. This will be how each cipher will be done.

Hill Cipher Background

The Hill Cipher utilizes either a 2x2 or 3x3 matrix. The matrix is filled with indexed values for the alphabetical string. The key is similarly indexed based upon the alphabet. The alphabet for the hill cipher is given below.

This alphabet is only the most standard one. The Hill cipher supports characters of any size if you update your mathematics appropriately. This Alphabet is all that's required for Radford CTFs and our own CTF. You may come across one that has included spaces to help make the plain-text more easily readable but it is not common.

Character Table

CHAR	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
NUM	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CHAR	Q	R	S	T	U	V	W	X	Y	Z						
NUM	16	17	18	19	20	21	22	23	24	25						

As you can see it uses the numbers 0 through 25 to encode the alphabet. Case also does not matter. How encryption works is that you take the input string and divide it into blocks of n characters where n is the length of your key you're using (either 4 or 9 characters). Then you put the input data after converting it into a number into a matrix and multiply it by your key modulus 26. Then decryption requires you to calculate the determinate of the ciphertext

The formula for encryption is given below.

$$\begin{pmatrix} key_a & key_b \\ key_c & key_d \end{pmatrix} \cdot \begin{pmatrix} msg_a & msg_b \\ msg_c & msg_d \end{pmatrix} \pmod{26} = \begin{pmatrix} ct_a & ct_b \\ ct_c & ct_d \end{pmatrix}$$

Where the message is broken up into 4 character blocks and then multiplied by the keyed matrix and then the value is calculated modulus 26.

The decryption key can be calculated via the following formula.

$$K_d = \det(K)^{-1} \cdot \text{adj}(K) \quad \text{where} \quad det = ad - bc \quad \text{and} \quad adj = \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

Then for decryption you do the same as you did for encryption but with K_d instead of K. Then you'll get the plain-text message back.

Messages have to be padded to an even number of characters so the letter X can be used as a padding character or any other to denote extraneous characters at the end. Remember that when doing it in chunks of 2x2 that you have to go top →bottom→right top→right bottom. This is so that the product matrix will map to the cipher-text correctly. Then you simply go in the same order to get the 4 characters out of the matrix in the correct order.

Hill Cipher Encryption

First you need to figure out what size of key you're going to be utilizing. For this paper we'll simply be using a 2x2 matrix. Then you need to select a key. Here we'll be using the key "HILL". You need to encode your key into a 2x2 matrix and then encrypt your plain text. You use the table given previously.

Key= "HILL"

1) Message= "HELLOWORLD"
2) $KEY = \begin{pmatrix} 7 & 11 \\ 8 & 11 \end{pmatrix}$
3) Encoding the message.
   A) Message = 7,4,11,11,14,22,14,17,11,3
4) Then multiply it by the key. So for the first 4 bytes it would be.
   A) $\begin{pmatrix} 7 & 11 \\ 8 & 11 \end{pmatrix} \cdot \begin{pmatrix} 7 & 11 \\ 4 & 11 \end{pmatrix} (\mathrm{mod}\, 26) = \begin{pmatrix} 15 & 16 \\ 22 & 1 \end{pmatrix}$
      I) Converting to letters it becomes. PWQB.
   B) Doing this for the rest of the message.
5) You end up with the numbers. 15,22,16,1,2,16,25,13,6,17
   A) Encoding to letters you get PWQBCQZNGR
6) Ciphertext=PWQDBCQZNGR

Hill Cipher Decryption

To get the inverted or decryption matrix you have to do the following.

Let K be the key matrix that we want to invert. For our cases it'll simply be a 2x2 matrix. So we have to first calculate K^{-1} such that $K * K^{-1} = 1 (\mathrm{mod}\ 26)$. The formula $K^{-1} = [d]^{-1} \cdot \mathrm{adj}(K)$ where $d \cdot [d]^{-1} = 1 (\mathrm{mod}\ 26)$ and adj is the adjugate of the matrix of K. and d is the determinate of K mod 26. We calculate the determinate d via the following formula. (ac-bd) mod 26. Then we multiply this

value by the adjugate of the matrix. After this we have the decryption key. To get the encryption key back we have to invert it again.

Given the key [7,11,8,11] we can calculate the decryption key through the following set.

1) $d=(7\cdot 11)-(8\cdot 11)$
   A) d=-11
2) $adj=\begin{pmatrix}11 & -11\\ -8 & 7\end{pmatrix}$
3) $k^{-1}=[-11]^{-1}\cdot\begin{pmatrix}11 & -11\\ -8 & 7\end{pmatrix} \bmod 26$
   A) thus we have to find the modular multiplicative inverse of the determinate. Because as you can see it's raised to the negative first power. Since the determinate is negative we have to use the extended euclidean algorithm extended to abstract algebraic fields.
   B) d is thus 7.
4) Thus we now have $k^{-1}=\begin{pmatrix}25 & 1\\ 22 & 23\end{pmatrix}$
5) The original cipher-text(just the start).
   A) $C=\begin{pmatrix}15 & 16\\ 22 & 1\end{pmatrix}$
6) Then we have to plug it back into our formula

   $$P=\begin{pmatrix}25 & 1\\ 22 & 23\end{pmatrix}\cdot\begin{pmatrix}15 & 16\\ 22 & 1\end{pmatrix} \bmod 26$$

   A) $\begin{pmatrix}397 & 401\\ 836 & 375\end{pmatrix}\bmod 26 \Rightarrow P=\begin{pmatrix}7 & 11\\ 4 & 11\end{pmatrix}$
7) Thus when decoding it we get back.
   A) HELL
   B) Which is the first four characters of the plain-text HELL
8) The decryption continues like this till the end.
   A) You then get the plain-text back of HELLOWORLD
9) Now the message is decrypted.

Hill Cipher Solver

The hill cipher uses a keyed 2x2 or 3x3 matrix to encrypt the data. You do this by first multiplying the plain-text matrix by the key matrix. To decrypt you get the inverted matrix modulus 26 and then do matrix multiplication of the cipher-text matrix and the key matrix modulus 26 to get back the plain text. You then multiply the inverse of the cipher-text matrix by the plain-text matrix modulus 26. This then gives you the decryption key.

Example cipher-text $\begin{pmatrix}15 & 16\\ 22 & 1\end{pmatrix}$ maps to $\begin{pmatrix}7 & 11\\ 4 & 11\end{pmatrix}$ We will utilize the following formula to calculate the encryption key. $P^{-1}=[d]^{-1}\cdot\text{adj}(P)$ Where $d\cdot d^{-1}=1\pmod{26}$

To solve for K(the encryption key) $[K]\cdot[P]=[C]\Rightarrow[K]=[C]\cdot[P]^{-1}$

To solve for D(decryption key). $[D]\cdot[C]=P\Rightarrow[D]=[P]\cdot[C]^{-1}$

1) First calculate the adjugate of P which is our ciphertext Given as $\text{adj}\left(\begin{bmatrix}7 & 11\\ 4 & 11\end{bmatrix}\right)$
   A) Using the formula above that $adj\left(\begin{bmatrix}a & b\\ c & d\end{bmatrix}\right)=\begin{bmatrix}d & -b\\ -c & a\end{bmatrix}$
   B) $adj\left(\begin{bmatrix}7 & 11\\ 4 & 11\end{bmatrix}\right)=\begin{bmatrix}11 & -11\\ -4 & 7\end{bmatrix}$
2) and d is the determinate of P.
   A) $d=ad-bc\pmod{26}$
   B) d=(7*11)-(4*11) mod 26
   C) d=7
   D) $d=[7]^{-1}\pmod{26}$
   E) modular multiplicative inverse of d is thus 15.
3) $15\cdot\begin{bmatrix}11 & -11\\ -4 & 7\end{bmatrix}\pmod{26}\Rightarrow P^{-1}=\begin{bmatrix}9 & 17\\ 18 & 1\end{bmatrix}$
4) $\begin{bmatrix}15 & 16\\ 22 & 1\end{bmatrix}.\begin{bmatrix}9 & 17\\ 18 & 1\end{bmatrix}\pmod{26}\Rightarrow K=\begin{bmatrix}7 & 11\\ 8 & 11\end{bmatrix}$
5) If instead we wanted the decryption key it'd be the same except that we would calculate C^{-1}
6) If we wanted to still utilize this encryption key we would calculate the modular multiplicative inverse of the matrix mod 26.

7) $K_d = \begin{pmatrix} 25 & 1 \\ 22 & 23 \end{pmatrix}$

Calculating the decryption key.

1) First calculate C^{-1}
   A) $det\left(\begin{bmatrix} 15 & 16 \\ 22 & 1 \end{bmatrix}\right) = -337 \bmod 26 \Rightarrow 1$
   B) We're lucky that the inverse of 1 is 1 here.
   C) $\text{adj}\left(\begin{bmatrix} 15 & 16 \\ 22 & 1 \end{bmatrix}\right) = \begin{bmatrix} 1 & -16 \\ -22 & 15 \end{bmatrix}$
   D) $1 \cdot \begin{bmatrix} 1 & -16 \\ -22 & 15 \end{bmatrix} \bmod 26 \Rightarrow [C]^{-1} = \begin{bmatrix} 1 & 10 \\ 4 & 15 \end{bmatrix}$
2) Then we multiply it by P.
   A) $K_d = \begin{bmatrix} 7 & 11 \\ 4 & 11 \end{bmatrix} \cdot \begin{bmatrix} 1 & 10 \\ 4 & 15 \end{bmatrix} \bmod 26 \Rightarrow K_d = \begin{bmatrix} 25 & 1 \\ 22 & 23 \end{bmatrix}$
   B) Thus our decryption key has been found.
3) Now we decrypt the original plain-text.
   A) We're going to do the first 4 characters.
   B) $\begin{bmatrix} 25 & 1 \\ 22 & 23 \end{bmatrix} \cdot \begin{bmatrix} 15 & 16 \\ 22 & 1 \end{bmatrix} \bmod 26 = \begin{bmatrix} 7 & 11 \\ 4 & 11 \end{bmatrix}$
4) Decoding it back to letters we get.
   A) 7=H, 4=E, 11=L
   B) HELL
5) This is the beginning of the plain-text as we knew originally thus we can decode the rest of the message accordingly.
   A) Message = 7,4,11,11,14,22,14,17,11,3
6) Then converting back to letters
   A) HELLOWORLD

Appendix B Source Code

#Hill Cipher Decrypter I used to solve Radford's flags but cleaned up.

```
def hill_decrypt(input_str,decryption_key):
    input_str_len=len(input_str);
    input_str_matricies=input_str_len//4;
    numeric=list( ord(input_str[i]) - 65 for i in
range(input_str_len))
    ct=list(list([[0,0],[0,0]]) for i in range(input_str_matricies))
    pt=list(list([[0,0],[0,0]]) for i in range(input_str_matricies))

    for i in range(input_str_matricies):
        ct[i][0][0]=numeric[i*4];ct[i][1][0]=numeric[i*4+1]
        ct[i][0][1]=numeric[i*4+2];ct[i][1][1]=numeric[i*4+3]

    for i in range(input_str_matricies):
        pt[i]=matrix_mul(decryption_key,ct[i],26)

    out=list([0] * input_str_len)
    for i in range(input_str_matricies):
        out[i*4]=chr(65+pt[i][0][0]);out[i*4+1]=chr(65+pt[i][1][0])
        out[i*4+2]=chr(65+pt[i][0][1]);out[i*4+3]=chr(65+pt[i][1][1])
    return ''.join(out)



'''
gcd calculator using Extended Euclidean Algorithm.

Python implementation of the extended euclidean algorithm for
calculating the gcd.
This code is the recursive variant as it is simpler.

'''
def gcd_fast(a,b):
    gcd=0;
```

```
    x=0;
    y=0;
    x=0
    #if a or b is zero return the other value and the coeffecient's
accordingly.
    if a==0:
        return (b,0,1)
    elif b==0:
        return (a,0,1)
    #otherwise actually perform the calculation.
    else:
        #set the gcd x and y according to the outputs of the function.
        # a is b (mod) a. b is just a.
        gcd, x, y = gcd_fast(b % a, a)
        #we're returning the gcd, x equals y - floor(b/a) * x
        # y is thus x.
        return (gcd, y - ( b // a ) * x, x)

# Calculates the modular multiplicative inverse of a and the modulus
value
# such that a * x = 1 % mod
# Also mod is the modulus.
# % is the modulus operator in python.
# This version is a generalization and improvement upon the standard
extended euclidean algorithm
# It is improved and works for all values a,m set Z.
# That means for all values of a and m that are real integers.
def mod_inv(a,mod):
    gcd=0;
    x=0;
    y=0;
    x=0;
    # if a is less than 0 do this.
    if a < 0:
        # if the modulus is less than zero
        # convert it to a positive value.
        #otherwise set the temporary variable x to the modulus.
```

```
        if mod < 0:
            x=-mod;
        else:
            x=mod;
        # while a is less than zero keep adding the abs value of the
modulus to it.
        while a < 0:
            a+=x
    #use the extended euclidean algorithm to calculate the gcd and
also bezout's coeffecients x and y.
    gcd, x, y = gcd_fast(a,mod)
    #I'm just viewing them to make sure that it is indeed working.
    print(gcd,x,y)
    #if the gcd is not 1 or -1 tell them that it's impossible to
invert.
    if gcd not in (-1,1):
        raise ValueError('Inputs are invalid. No modular
multiplicative inverse exists between {} and {} gcd:{}.\
n'.format(a,mod,gcd))
    #otherwise do the inversion.
    else:
        #if m is negative do the following.
        if gcd == -1:
            #if x is less than zero convert x to positive and add it
to the modulus.
            if x < 0:
                return mod - x
            #otherwise just add x to the modulus.
            else:
                return x + mod
        #otherwise if m is both positive return x (mod m)
        else:
            return x % mod
# This function calculates the determinate of A
# via the following formula.
# (a*d) - (b*c)
def det(A):
```

```
    d=(A[0][0]*A[1][1])-(A[0][1]*A[1][0])
    return d

# This calculates the inverse of the determinant of A and m.
def inv_det(A,m):
    return mod_inv(det(A),m)

# This function calculates the adjugate of A
# The formula is [[a,b],[c,d]] => [[d,-b],[-c,a]]
def adj(A):
    B=[[0,0],[0,0]]
    B[0][0]=A[1][1]
    B[0][1]=-A[0][1]
    B[1][0]=-A[1][0]
    B[1][1]=A[0][0]
    return B

# This function calculates the inverse of the matrix mod 26.
def matrix_inv(A,m):

    A_inv=[[0,0],[0,0]]
    d=inv_det(A,m)
    a=adj(A)
    A_inv[0][0]=d*a[0][0] % m
    A_inv[0][1]=d*a[0][1] % m
    A_inv[1][0]=d*a[1][0] % m
    A_inv[1][1]=d*a[1][1] % m

    return A_inv

# This function multiplies two matricies A and B and it takes an
optional third argument
# m which will take the results and apply a modulus on their
matricies. If it is not set
# it will return the normal multiplication.
# ex. matrix_multiply(A,B,m) will return the results of [A]*[B] % m
```

```
# If the second variable B is not a matrix then it multiplies A by
that value.
# This is not the normal method but it keeps it simplified.
def matrix_mul(A,B,m=0):
    C=[[0,0],[0,0]]
    if len(A) != 2 and len(A[0]) != 2:
        raise ValueError('Your matrix must be a 2x2');
    if type(B) == list:
        if len(A) == len(B):
            raise ValueError('Both matricies must be 2x2');
        C[0][0] = (A[0][0]*B[0][0])+(A[0][1]*B[1][0])
        C[0][1] = (A[0][0]*B[0][1])+(A[0][1]*B[1][1])
        C[1][0] = (A[1][0]*B[0][0])+(A[1][1]*B[1][0])
        C[1][1] = (A[1][0]*B[0][1])+(A[1][1]*B[1][1])
    else:
        C[0][0] = A[0][0]*B
        C[0][1] = A[0][1]*B
        C[1][0] = A[1][0]*B
        C[1][1] = A[1][1]*B

    if m != 0:
       C[0][0] = C[0][0] % m
       C[0][1] = C[0][1] % m
       C[1][0] = C[1][0] % m
       C[1][1] = C[1][1] % m

    return C
```

Appendix C Real World Attacks

Below you will find the information applied to real attacks against actual CTF challenges I solved. Plus it will also show the same basic premise applied to a 3x3 Hill Cipher.

Radford Hill Cipher #1

Introduction

Suppose the ciphertext

FXYTQ YSTRO UGJXD XUBCC SYOEC GLOUV MCUKT EZXGV ROCM

was encrypted with a Hill cipher using the key matrix

3 5 11 18

Decipher the message.

The Key Matrix is $\text{key}=\begin{bmatrix}3 & 5\\ 11 & 18\end{bmatrix}$

So to decrypt it we need to find the decryption key by inverting the key.

1. First take the matrix $\text{key}=\begin{bmatrix}3 & 5\\ 11 & 18\end{bmatrix}$
2. Then to invert it we have to calculate the determinant and also the adjugate of the matrix then multiply it by the original key matrix.
   A. $\det(key)=((3*18)-(5*11))\bmod 26$
      1. det=(54-55) mod 26
      2. det=-1 mod 26
      3. det=25
   B. Now we have to calculate the modular multiplicative inverse of the determinant.
      1. $[25]^{-1}\bmod 26\Rightarrow 25$ mat
      2. Thus our inverse determinant is 25.
   C. $\text{Adj}(key)=\begin{bmatrix}18 & -5\\ -11 & 3\end{bmatrix}$

D. $key^{-1} = 25 \cdot \begin{bmatrix} 18 & -5 \\ -11 & 3 \end{bmatrix} \bmod 26$

1. $25 \cdot \begin{pmatrix} 18 & -5 \\ -11 & 3 \end{pmatrix} = \begin{pmatrix} 450 & -125 \\ -275 & 75 \end{pmatrix}$

2. $\begin{pmatrix} 450 & -125 \\ -275 & 75 \end{pmatrix} \bmod 26 = \begin{pmatrix} 8 & 5 \\ 11 & 23 \end{pmatrix}$

E. $key^{-1} = \begin{pmatrix} 8 & 5 \\ 11 & 23 \end{pmatrix}$

You now have the decryption key which is

$$\begin{pmatrix} 8 & 5 \\ 11 & 23 \end{pmatrix}$$

You then apply the matrix transformations in the same way that you did when encrypting but with the decryption key instead and get the flag below.

Flag:HILL CIPHERS ENCRYPT MESSAGES IN BLOCKS OF LETTERS(A) which the last A isn’t included in the actual flag for reasons. Also he wants it to be separated into words.

Radford Hill Cipher #2

Introduction

Suppose you intercept the ciphertext

ZAGXE NGUFO HMGXE NOAOW IRFCQ NOYAF FPBDF CSXMU

which was known to be enciphered by a key matrix using a Hill cipher. Suppose you somehow guess that the first four letters of the ciphertext resulted 7from the plaintext letter crib given by FIND. Recover the key matrix and decipher the message.

This one requires you to use the solver math from up above. You're going to have to remember the formulas and calculate the decryption key so that you can decipher it.

Starting the Attack

First you have to take the first 4 characters of the cipher-text ZAGX and encode the vectors into a matrix. Then you do the same with the crib FIND.

$$\text{ZAGX}=\begin{bmatrix}25 & 6\\ 0 & 23\end{bmatrix}$$

$$\text{FIND}=\begin{bmatrix}5 & 13\\ 8 & 3\end{bmatrix}$$

Then we're going to actually carry out the attack. From now on only the steps will be shown. It expects you are familiar with the math behind it.

Recall that $K_d=[P]\cdot[C]^{-1}$

1) Calculate the modular multiplicative inverse of C.
   A) Recall that the modular multiplicative inverse of the array C is $\det[C]^{-1}\cdot\text{adj}[C]$
   B) $\text{adj}[C]=\begin{bmatrix}23 & -6\\ 0 & 25\end{bmatrix}$
   C) $\det[C]=(25*23)-(6*0)\Rightarrow\det[C]=575$
      I) $[575]^{-1}\bmod 26=9$
   D) $[C]^{-1}=9\cdot\begin{bmatrix}23 & -6\\ 0 & 25\end{bmatrix}\bmod 26\Rightarrow[C]^{-1}=\begin{bmatrix}25 & 24\\ 0 & 17\end{bmatrix}$

2) $\begin{bmatrix}5 & 13\\ 8 & 3\end{bmatrix}.\begin{bmatrix}25 & 24\\ 0 & 17\end{bmatrix} mod\, 26 = \begin{bmatrix}21 & 3\\ 18 & 9\end{bmatrix}$
   A) Now we have the decryption key D or Key_d
3) Now multiply the decryption key by the cipher text to decrypt it.
   A) $[P] = \begin{bmatrix}21 & 3\\ 18 & 9\end{bmatrix}.\begin{bmatrix}25 & 6\\ 0 & 23\end{bmatrix} \bmod 26 \Rightarrow [P] = \begin{bmatrix}5 & 13\\ 8 & 3\end{bmatrix}$
   B) We have the same plain-text back again thus we have found the proper decryption key.
4) Continue on with the same decryption matrix applying it to the cipher-text till you get the plain-text back.
   A) FINDTHECRIBANDTHISWILLHELPCAPTURETHEFLAG
      I) Once again we have to actually add spaces for whatever reason.
   B) Thus the flag has spaces added.
5) FIND THE CRIB AND THIS WILL HELP CAPTURE THE FLAG
6) You now have the flag.

The Attack is complete and you can see how I solved the flag. The source code is included in the other appendix.