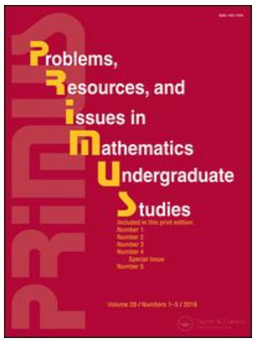
An Algorithm for a Matrix-Based Enigma Encoder from a Variation of the Hill Cipher as an Application of 2 × 2 Matrices

Article *in* PRIMUS · August 2018

DOI: 10.1080/10511970.2018.1493010

Porter Coggins
Bemidji State University
24 PUBLICATIONS **18** CITATIONS



PRIMUS

Problems, Resources, and Issues in Mathematics Undergraduate Studies

ISSN: 1051-1970 (Print) 1935-4053 (Online) Journal homepage: https://www.tandfonline.com/loi/upri20

An Algorithm for a Matrix-Based Enigma Encoder from a Variation of the Hill Cipher as an Application of 2 × 2 Matrices

Porter E. Coggins III & Tim Glatzer

To cite this article: Porter E. Coggins III & Tim Glatzer (2019): An Algorithm for a Matrix-Based Enigma Encoder from a Variation of the Hill Cipher as an Application of 2 × 2 Matrices, PRIMUS, DOI: 10.1080/10511970.2018.1493010

To link to this article: https://doi.org/10.1080/10511970.2018.1493010

Accepted author version posted online: 15 Aug 2018.
Published online: 01 Apr 2019.

PRIMUS, 0(0): 1–18, 2019
Copyright © Taylor & Francis Group, LLC
ISSN: 1051-1970 print / 1935-4053 online
DOI: 10.1080/10511970.2018.1493010

An Algorithm for a Matrix-Based Enigma Encoder from a Variation of the Hill Cipher as an Application of 2 × 2 Matrices

Porter E. Coggins III and Tim Glatzer

Abstract: We present an algorithm for a matrix-based Enigma-type encoder based on a variation of the Hill Cipher as an application of 2 × 2 matrices. In particular, students will use vector addition and 2 × 2 matrix multiplication by column vectors to simulate a matrix version of the German Enigma Encoding Machine as a basic example of cryptography. The ideal assumed background is a rudimentary familiarity of matrix multiplication and vector addition, but students who have successfully completed introductory linear algebra, number theory, and discrete mathematics will find this example accessible. Making this connection between these two systems provides a rich field for the introduction of the concepts mentioned above.

Keywords: Linear algebra - matrix theory, modular inverse matrices, cryptography, Hill Cipher, M4 German Enigma Machine, ZDM Classification D40, M10, R

1. INTRODUCTION

We present an algorithm for a matrix-based Enigma Encoder developed from a variation of the Hill Cipher that can be used as an example of an application of 2 × 2 matrices during or after completion of the first chapter of a typical undergraduate course in Linear Algebra - Matrix Theory and as a connection point to other courses and disciplines. We will provide very brief introductions to both a typical Enigma and the Hill Cipher with references for further reading before providing details of an

Address all correspondence to Dr. Porter E. Coggins III, Professional Education, Bemidji State University, Bemidji, MN 56601. E-mail: porter.coggins@bemidjistate.edu
Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/upri.

algorithm based on them, a plaintext message encrypted with this new algorithm, and possible classroom connections. The algorithm presented here uses only 2×2 matrices and can be inserted into the curriculum of courses on matrix theory and linear algebra, number theory, discrete mathematics, coding, and introductory cryptography.

In 1929 [9] and 1931 [10] the American mathematician Lester Hill defined a polygraphic substitution encryption scheme based on a linear transformation of plaintext characters into ciphertext characters. The German military used Enigma machines during World War II to send secure messages [21, 24]. Variations of the military encoding machine came to be known collectively as *Enigma*, *Enigma Machines, German Enigma Machines,* or *Enigma Coding Machines*. Considerable time and effort was spent by various working groups in the Allied Forces analyzing cipher-text from Enigma transmissions during the war in attempts to decipher the messages. Section 5 at the end of this paper states references regarding the history, historical significance, and continued research on both Enigma Machines and the Hill Cipher.

1.1. Overview of a Typical German Enigma Machine

Prior to World War II, businesses were using encoding machines of various sorts for secure communications [12]. The German military adapted a version of an encoding machine to encrypt and decrypt military messages. Physically, a typical German Enigma Machine is an electro-mechanical machine about the size of a typewriter and housed in a solid oak cabinet. Figure 1 shows an actual Enigma Machine and Figure 2 represents a typical encryption path from entering a plaintext character to the output of the ciphertext character.

1.2. Overview of the Hill Cipher

Although the details of the Hill cipher can be readily found (see Section 5), it may be surprising to learn that it has continued to generate research interest from shortly after Hill's original publication up to the present. For a very brief example using Hill's technique, we will use a set of 29 characters (see Table 1) and perform all arithmetic calculations in modulo 29. We add punctuation characters *space*, *period*, and *comma* to the English alphabet to help avoid ambiguity in the plaintext. As 29 is prime, all matrices with non-zero determinants will have a modular inverse [18, p. 970]. We limit ourselves to 2×2 matrices because this small-order matrix affords an introduction to a number of topics that can be easily computed by hand, and leads to conceptual building blocks for generalization of these topics to larger-order matrices [8]. Before giving the

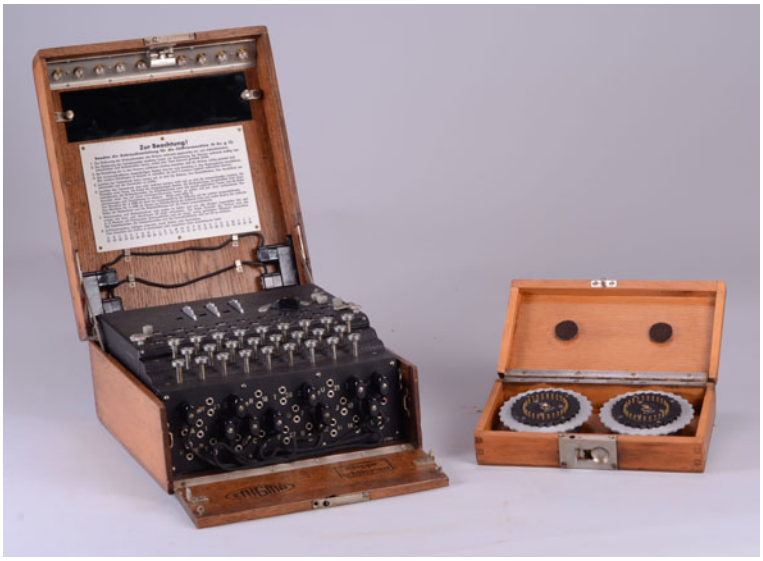


Figure 1. A three-Rotor German Enigma Machine (image used with permission from copyright holder www.EnigmaMuseum.com).

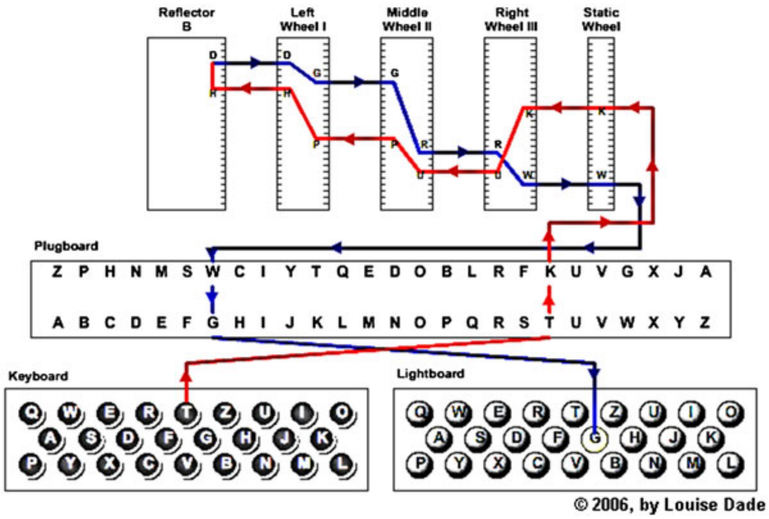


Figure 2. How one letter is changed into another letter at each stage as it passes through an Enigma Machine. (image used with permission of the copyright holder [3]).

Table 1. Character – Numerical Assignment map

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	.	,	_
26	6	11	12	5	24	0	16	18	23	28	9	27	22	2	15	20	1	14	7	10	8	19	21	13	3	17	4	25

details of our algorithm of a matrix-based Enigma explained below, we applied the Hill Cipher to the plaintext *meet at the cafe at midnight.* (with the period "." included in the plaintext message and blank characters for padding to create four-character blocks) using the involutory key matrix

$$\begin{bmatrix} 4 & 28 \\ 15 & 25 \end{bmatrix}$$

and character map in Table 1.

To encrypt, we will first construct ordered pairs of letters noting that our character set includes the additional punctuation characters "_", ".", and "," (representing *space*, *period*, and *comma*), and agree to add, or *pad*, with blank spaces (numerical value 25 in our Table 1 above) to create an output of pairs of ordered pairs: (27,5), (5,7), (25,26), (7,25), (7,16), (5,25), (11,26), (24,5), (25,26), (7,25), (27,18), (12,22), (18,0), (16,7), (17,25), (25,25).

Second, each ordered pair above is expressed as a 2 × 1 plaintext column vector:

$$\begin{bmatrix} 27 \\ 5 \end{bmatrix} \begin{bmatrix} 5 \\ 7 \end{bmatrix} \begin{bmatrix} 25 \\ 26 \end{bmatrix} \begin{bmatrix} 7 \\ 25 \end{bmatrix} \begin{bmatrix} 7 \\ 16 \end{bmatrix} \begin{bmatrix} 5 \\ 25 \end{bmatrix} \begin{bmatrix} 11 \\ 26 \end{bmatrix} \begin{bmatrix} 24 \\ 5 \end{bmatrix} \begin{bmatrix} 25 \\ 26 \end{bmatrix}$$

$$\begin{bmatrix} 7 \\ 25 \end{bmatrix} \begin{bmatrix} 27 \\ 18 \end{bmatrix} \begin{bmatrix} 12 \\ 22 \end{bmatrix} \begin{bmatrix} 18 \\ 0 \end{bmatrix} \begin{bmatrix} 16 \\ 7 \end{bmatrix} \begin{bmatrix} 17 \\ 25 \end{bmatrix} \begin{bmatrix} 25 \\ 25 \end{bmatrix}$$

Third, each 2 × 1 column vector is multiplied (mod 29) with the involutory matrix

$$A = \begin{bmatrix} 4 & 28 \\ 15 & 25 \end{bmatrix}$$

to generate the ciphertext vector. Since all products are calculated mod 29, we will not write "mod 29" after calculations.

$$\begin{bmatrix} 4 & 28 \\ 15 & 25 \end{bmatrix} \begin{bmatrix} 27 \\ 5 \end{bmatrix} = \begin{bmatrix} 16 \\ 8 \end{bmatrix}, \begin{bmatrix} 4 & 28 \\ 15 & 25 \end{bmatrix} \begin{bmatrix} 5 \\ 7 \end{bmatrix} = \begin{bmatrix} 13 \\ 18 \end{bmatrix}, \begin{bmatrix} 4 & 28 \\ 15 & 25 \end{bmatrix} \begin{bmatrix} 25 \\ 26 \end{bmatrix}$$

$$= \begin{bmatrix} 16 \\ 10 \end{bmatrix}, \begin{bmatrix} 4 & 28 \\ 15 & 25 \end{bmatrix} \begin{bmatrix} 7 \\ 25 \end{bmatrix} = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$$

Continuing the computations, the complete list of the encrypted column vectors, including those listed above, are

$$\begin{bmatrix}16\\8\end{bmatrix}\begin{bmatrix}13\\18\end{bmatrix}\begin{bmatrix}16\\10\end{bmatrix}\begin{bmatrix}3\\5\end{bmatrix}\begin{bmatrix}12\\12\end{bmatrix}\begin{bmatrix}24\\4\end{bmatrix}\begin{bmatrix}18\\3\end{bmatrix}\begin{bmatrix}4\\21\end{bmatrix}\begin{bmatrix}16\\10\end{bmatrix}\begin{bmatrix}3\\5\end{bmatrix}$$

$$\begin{bmatrix}3\\14\end{bmatrix}\begin{bmatrix}26\\5\end{bmatrix}\begin{bmatrix}14\\9\end{bmatrix}\begin{bmatrix}28\\9\end{bmatrix}\begin{bmatrix}14\\10\end{bmatrix}\begin{bmatrix}17\\14\end{bmatrix}$$

with the corresponding ordered pairs (16,8) (13,18) (16,10) (3,5) (12,12) (24,4) (18,3) (4,21) (16,10) (3,5) (3,14) (26,5) (14,9) (28,9) (14,10) (17,14) respectively.

Finally, the numbers from the ordered pairs are mapped to the letter characters from the character map, Table 1, produces the following string of characters in four – character blocks: generating the ciphertext: *hvyi huze ddf, iz,x huze zsae slkl su.s*

2. ANALOGY OF THE HILL CIPHER EXTENSION ALGORITHM TO A GERMAN ENIGMA MACHINE

We will use 2 × 2 matrices to represent the rotor wheels of a typical German Enigma Machine and refer to these key matrices as *rotor matrices*. However, unlike the plaintext signal flow of an Enigma, which passes through each character rotor wheel, the reflector, and back through the rotors again in reverse order (see Figure 2), the algorithm defined in this paper passes the plaintext in only a single direction and without an analog of the reflector wheel for simplicity. The novel variation to the Hill cipher that is presented here is to construct a series of invertible matrices, in which plaintext characters will multiply through as the matrix values *rotate* position in each matrix in a determined order based on counter values and matrix element orientation. We will keep track of the counter and orientation of each rotor matrix in a state 5-tuple explained below. This process will be similar to (but not exactly like) the rotation of the character rotor wheels in a German Enigma Machine. We will mimic the Enigma plugboard with a random column vector with entries raised to an exponent. The state 5-tuple will be used to set the rotor matrices to the correct positions for decryption. After stating basic definitions, we encrypt the message used in the example above and compare the encrypted message of this variation with that of the Hill Cipher given above.

2.1. Definitions

Our basic definitions include p is prime, $\mathbb{Z}_p$ is the ring of integers modulo p (with all calculations in modulo p), m is the length of the plaintext, and K represents the key space of invertible matrices modulo p. For an abstract representation, let ${}_iR_{r_q}$ represent a 2×2 *rotor matrix*

$$\begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix},$$

where the left subscript on ${}_iR_{r_q}$ is a counter that indicates the ith plaintext or cipher-ordered pair character values being processed, and the right subscript r represents the rotor matrix (numbered 0 through 3 for the four matrix rotors used in this paper). The subscripts q to r, designated as $h, j, k, l,$ represent counters for tracking the state of the matrix rotors. The left subscript i, defined above, will also be used as a counter in the state 5-tuple. Each rotor matrix will start in an initial state "0" as the first rotor position with an orientation as

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

with the following canonical matrix value assignments: $a = x_{1,1}$, $b = x_{1,2}$, $c = x_{2,1}$, and $d = x_{2,2}$. *Rotating* matrix values will be represented using the following convention:

A single clockwise rotation ($1 \times 90°$) of a rotor matrix a general position

$$\begin{bmatrix} u & v \\ w & x \end{bmatrix}$$

results in the new matrix rotor position

$$\begin{bmatrix} w & u \\ x & v \end{bmatrix}.$$

Suppose the "0" state is:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

Each following state corresponds to a clockwise rotation of the previous state with the remaining three states represented as

$$\begin{bmatrix} c & a \\ d & b \end{bmatrix}, \begin{bmatrix} d & c \\ b & a \end{bmatrix}, \text{and} \begin{bmatrix} b & d \\ a & c \end{bmatrix}.$$

Let σ_g represent the *state* of the orientation of the gth rotor matrix. Let the 5-tuple $(i, \sigma_3, \sigma_2, \sigma_1, \sigma_0)$ represent the ith state of all rotor matrices at count i ranging from 0 to $m/2$ for an even number of plaintext characters m, and $(m+y)/2$ for an odd number of plaintext characters m with padded characters appended to the end of the plaintext message to create an blocks of four characters each. We must keep track of the direction of rotation of the rotor matrix values, noting that we will perform clockwise matrix rotor value rotations for encryption and counterclockwise rotations of inverse rotor matrix values for decryption. We will represent the direction of matrix values rotation by an arrow over a matrix: $\vec{R}$ for a clockwise rotation of matrix values, and $\overleftarrow{R}$ for a counter-clockwise rotation of matrix values. The arrow-hat will be omitted when not relevant for the discussion.

Let a *cycle* represent a plain (cipher) text path through the first plugboard vector, through all 2 × 2 rotor matrices to completion of encryption (decryption). Plaintext characters will be represented by the Latin alphabet and ciphertext characters by the Greek alphabet with a left superscript representing the character counter over the length of the plaintext (encryption) or ciphertext (decryption), and consecutive pairs of characters will be multiplied by appropriate rotations of a rotor matrix (encryption) or inverse matrix (decryption).

Let

$$ {}_i\vec{R}_{r_q} \begin{bmatrix} {}^{s}a \\ {}^{s+1}a \end{bmatrix} $$

represent the matrix product

$$ \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} \begin{bmatrix} {}^{s}a \\ {}^{s+1}a \end{bmatrix} $$

of the 2 × 2 rotor matrix ${}_i\vec{R}_{r_q}$ and a 2 × 1 plaintext character vector

$$ \begin{bmatrix} {}^{s}a \\ {}^{s+1}a \end{bmatrix} $$

where ${}^{s}a$ and ${}^{s+1}a$ represent consecutive plaintext characters. Let

$$ \begin{bmatrix} {u_0}^{i} \\ {u_1}^{i} \end{bmatrix} $$

represent an initial *plugboard* 2 × 1 vector raised to the counter ith power (modulo p) that is added to each plaintext message character pair prior to multiplication with the first matrix rotor, where i is the value of the state of the character at the start of the character encryption (decryption) cycle and the index of the character pair. This vector is somewhat analogous to the Enigma plugboard, but because the values change by the exponent of the counter value, it changes during encryption (decryption) to increase diffusion [27].

2.2. General Form of the Algorithm

After a brief overview of the algorithm, an example will be given. The plaintext message character pair is conditioned by first adding the plugboard vector to the plaintext message character pair

$$\begin{bmatrix} {}^{s}a \\ {}^{s+1}a \end{bmatrix} + \begin{bmatrix} u^{i}{}_{0} \\ u^{i}{}_{1} \end{bmatrix} = \begin{bmatrix} {}^{s}\lambda \\ {}^{s+1}\lambda \end{bmatrix}.$$

The resulting vector sum is then multiplied with the first rotor matrix

$${}_{i}\vec{R}_{0_h} \begin{bmatrix} {}^{s}\lambda \\ {}^{s+1}\lambda \end{bmatrix} = \begin{bmatrix} {}^{s}\beta \\ {}^{s+1}\beta \end{bmatrix}$$

and the state of the first rotor matrix is updated from h to $h+1$. If $i \equiv 0(\mathrm{mod}4)$, update the state of the second matrix rotor and let $j = j+1$. If $i \equiv 0$ (mod16), update the state of the third matrix rotor and let $k = k+1$. If $i \equiv 0$ (mod64), update the state of the fourth matrix rotor and let $l = l+1$. Multiply the output vector from above with the second matrix rotor

$${}_{i}\vec{R}_{1_j} \begin{bmatrix} {}^{s}\beta \\ {}^{s+1}\beta \end{bmatrix} = \begin{bmatrix} {}^{s}\gamma \\ {}^{s+1}\gamma \end{bmatrix},$$

multiply the output vector from above with the third matrix rotor

$${}_{i}\vec{R}_{2_k} \begin{bmatrix} {}^{s}\gamma \\ {}^{s+1}\gamma \end{bmatrix} = \begin{bmatrix} {}^{s}\delta \\ {}^{s+1}\delta \end{bmatrix},$$

multiply the output vector from above with the fourth matrix rotor

$${}_{i}\vec{R}_{3_l} \begin{bmatrix} {}^{s}\delta \\ {}^{s+1}\delta \end{bmatrix} = \begin{bmatrix} {}^{s}\varsigma \\ {}^{s+1}\varsigma \end{bmatrix} = \begin{bmatrix} {}^{s}\varepsilon \\ {}^{s+1}\varepsilon \end{bmatrix}$$

with the encryption of the character pair $({}^{s}a, {}^{s+1}a)$ as $({}^{s}\varepsilon, {}^{s+1}\varepsilon)$.

2.3. Initialization of the Linear Combination of Matrix Rotors

Hill [9] originally used *involutory* matrices (which are inverses of themselves), but the key space of involutory matrices is smaller than the space of invertible matrices [20, p. 12]. Therefore, using invertible matrices that are not involutory increases the number of key matrices, and hence, variability in the encryption process. Furthermore, knowing the encryption key matrices does not automatically identify the decryption key matrices. For that reason, we define four invertible, but not involutory, matrices to represent each rotor matrix analogous to the character rotor wheels of a four-wheel German Enigma Machine.

We will use the character set from Table 1 of the Hill Cipher example above with $p = 29$. For each rotor matrix we will choose values *a, b, c*, and *d* in

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

such that for matrix A with prime order p,

$$A^{-1} = (\det A)^{-1} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix},$$

the modular inverse of

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

exists mod p [18, p. 970; 25, pp. 175–179; 28, p. 17]. For the equivalent of plugboard character swapping assignments, we will use a non-zero 2×1 column vector raised to powers based on the number of plaintext pairs. Let us arbitrarily choose the initial plugboard matrix

$$\begin{bmatrix} 20^i \\ 12^i \end{bmatrix}$$

where i ranges from zero to the number of message pairs as explained above. We will keep track of rotor matrix rotation states (Table 2) and list the rotor matrices in the table in reverse order (in descending order left to right) to keep track of rotor matrix states. This representation of states in the standard place-value notation in base 4 is used to track all rotor states and text pairs for decryption.

We will use the character set as shown in the Hill Cipher Table 1 example above.

For our example, we will use four rotor matrices (Table 3) and the same plaintext as in the Hill Cipher example above, *meet at the cafe at midnight.* Proceeding as in the Hill Cipher example above, we create ordered pairs of consecutive message characters with padding: (27,5) (5,7) (25,26) (7,25) (7,16) (5,25) (11,26) (24,5) (25,26) (7,25) (27,18) (12,22) (18,0) (16,7) (17,25) (25,25) and form 2×1 column vectors from the ordered pairs above:

$$\begin{bmatrix} 27 \\ 5 \end{bmatrix} \begin{bmatrix} 5 \\ 7 \end{bmatrix} \begin{bmatrix} 25 \\ 26 \end{bmatrix} \begin{bmatrix} 7 \\ 25 \end{bmatrix} \begin{bmatrix} 7 \\ 16 \end{bmatrix} \begin{bmatrix} 5 \\ 25 \end{bmatrix} \begin{bmatrix} 11 \\ 26 \end{bmatrix} \begin{bmatrix} 24 \\ 5 \end{bmatrix} \begin{bmatrix} 25 \\ 26 \end{bmatrix} \begin{bmatrix} 7 \\ 25 \end{bmatrix} \begin{bmatrix} 27 \\ 18 \end{bmatrix} \begin{bmatrix} 12 \\ 22 \end{bmatrix} \begin{bmatrix} 18 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 16 \\ 7 \end{bmatrix} \begin{bmatrix} 17 \\ 25 \end{bmatrix} \begin{bmatrix} 25 \\ 25 \end{bmatrix}$$

Table 2. State Table for Rotor Matrix Position with Respect to Index i

i	Fourth Rotor Matrix R_3, State 4^3 Place in Base 4	Third Rotor Matrix R_2, State 4^2 Place in Base 4	Second Rotor Matrix R_1, State 4^1 Place in Base 4	First Rotor Matrix, R_0, State 4^0 Place in Base 4
0	0	0	0	0
1	0	0	0	1
2	0	0	0	2
3	0	0	0	3
4	0	0	1	0
⋮	⋮	⋮	⋮	⋮
10	0	0	2	2
⋮	⋮	⋮	⋮	⋮
22	0	2	1	2
⋮	⋮	⋮	⋮	⋮
31	0	1	3	3
⋮	⋮	⋮	⋮	⋮
63	0	3	3	3
64	1	0	0	0
⋮	⋮	⋮	⋮	⋮

Now, we will step through the encryption algorithm for the first character pair represented as the 2×1 column vector

$$\begin{bmatrix} 27 \\ 5 \end{bmatrix}.$$

Set $i = 0$. Add the plugboard column vector to the first plaintext vector:

$$\begin{bmatrix} 20^0 \\ 12^0 \end{bmatrix} + \begin{bmatrix} 27 \\ 5 \end{bmatrix} = \begin{bmatrix} 28 \\ 6 \end{bmatrix}.$$

Multiply the output above with the first rotor matrix R_0 in the initial position

$$\begin{bmatrix} 8 & 17 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} 28 \\ 6 \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \end{bmatrix}$$

and rotate the first rotor matrix values one quarter turn clockwise. Multiply the output above with the second rotor matrix in the initial position

$$\begin{bmatrix} 27 & 23 \\ 19 & 8 \end{bmatrix} \begin{bmatrix} 7 \\ 0 \end{bmatrix} = \begin{bmatrix} 15 \\ 17 \end{bmatrix}.$$

Table 3. Example rotor matrices and modular inverses (mod 29)

	Orientation Position 0	Orientation Position 1	Orientation Position 2	Orientation Position 3
Rotor Matrix 0	$\begin{bmatrix} 8 & 17 \\ 1 & 5 \end{bmatrix}$	$\begin{bmatrix} 1 & 8 \\ 5 & 17 \end{bmatrix}$	$\begin{bmatrix} 5 & 1 \\ 17 & 8 \end{bmatrix}$	$\begin{bmatrix} 17 & 5 \\ 8 & 1 \end{bmatrix}$
Modular Inverse Rotor Matrix 0	$\begin{bmatrix} 4 & 27 \\ 5 & 18 \end{bmatrix}$	$\begin{bmatrix} 27 & 18 \\ 4 & 5 \end{bmatrix}$	$\begin{bmatrix} 18 & 5 \\ 27 & 4 \end{bmatrix}$	$\begin{bmatrix} 5 & 4 \\ 18 & 27 \end{bmatrix}$
Rotor Matrix 1	$\begin{bmatrix} 27 & 23 \\ 19 & 8 \end{bmatrix}$	$\begin{bmatrix} 19 & 27 \\ 8 & 23 \end{bmatrix}$	$\begin{bmatrix} 8 & 19 \\ 23 & 27 \end{bmatrix}$	$\begin{bmatrix} 23 & 8 \\ 27 & 19 \end{bmatrix}$
Modular Inverse Rotor Matrix 1	$\begin{bmatrix} 6 & 19 \\ 22 & 13 \end{bmatrix}$	$\begin{bmatrix} 19 & 13 \\ 6 & 22 \end{bmatrix}$	$\begin{bmatrix} 13 & 22 \\ 19 & 6 \end{bmatrix}$	$\begin{bmatrix} 22 & 6 \\ 13 & 19 \end{bmatrix}$
Rotor Matrix 2	$\begin{bmatrix} 12 & 17 \\ 25 & 7 \end{bmatrix}$	$\begin{bmatrix} 25 & 12 \\ 7 & 17 \end{bmatrix}$	$\begin{bmatrix} 7 & 25 \\ 17 & 12 \end{bmatrix}$	$\begin{bmatrix} 17 & 7 \\ 12 & 25 \end{bmatrix}$
Modular Inverse Rotor Matrix 2	$\begin{bmatrix} 1 & 10 \\ 13 & 10 \end{bmatrix}$	$\begin{bmatrix} 10 & 10 \\ 1 & 13 \end{bmatrix}$	$\begin{bmatrix} 10 & 13 \\ 10 & 1 \end{bmatrix}$	$\begin{bmatrix} 13 & 1 \\ 10 & 10 \end{bmatrix}$
Rotor Matrix 3	$\begin{bmatrix} 15 & 18 \\ 16 & 11 \end{bmatrix}$	$\begin{bmatrix} 16 & 15 \\ 11 & 18 \end{bmatrix}$	$\begin{bmatrix} 11 & 16 \\ 18 & 15 \end{bmatrix}$	$\begin{bmatrix} 18 & 11 \\ 15 & 16 \end{bmatrix}$
Modular Inverse Rotor Matrix 3	$\begin{bmatrix} 15 & 15 \\ 23 & 2 \end{bmatrix}$	$\begin{bmatrix} 15 & 2 \\ 15 & 23 \end{bmatrix}$	$\begin{bmatrix} 2 & 23 \\ 15 & 15 \end{bmatrix}$	$\begin{bmatrix} 23 & 15 \\ 2 & 15 \end{bmatrix}$

Multiply the output above with the third rotor matrix in the initial position

$$\begin{bmatrix} 12 & 17 \\ 25 & 7 \end{bmatrix}\begin{bmatrix} 15 \\ 17 \end{bmatrix} = \begin{bmatrix} 5 \\ 1 \end{bmatrix}.$$

Multiply the output above with the fourth rotor matrix in the initial position

$$\begin{bmatrix} 15 & 18 \\ 16 & 11 \end{bmatrix}\begin{bmatrix} 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 4 \end{bmatrix}.$$

This corresponds to the alphabet ciphertext characters *b,* using Table 1 above. Increment i to $i = 1$, update the plugboard vector to

$$\begin{bmatrix} 20^1 \\ 12^1 \end{bmatrix}$$

and add the updated plugboard vector to the next 2×2 plaintext vector that represents the next ordered pair. We proceed as above keeping in mind that we rotate matrix rotor 0 after each argument pass, and rotate rotor matrix 1 one-quarter turn clockwise when $i \equiv 0 \pmod{4}$, rotate rotor matrix 2 – one-quarter turn clockwise when $i \equiv 0 \pmod{16}$, and rotate matrix 3 – one-quarter turn clockwise when $i \equiv 0 \pmod{64}$. Continuing the algorithm with the remaining vectors transforms the original 2×1 plaintext vectors:

$$\begin{bmatrix}27\\5\end{bmatrix}\begin{bmatrix}5\\7\end{bmatrix}\begin{bmatrix}25\\26\end{bmatrix}\begin{bmatrix}7\\25\end{bmatrix}\begin{bmatrix}7\\16\end{bmatrix}\begin{bmatrix}5\\25\end{bmatrix}\begin{bmatrix}11\\26\end{bmatrix}\begin{bmatrix}24\\5\end{bmatrix}\begin{bmatrix}25\\26\end{bmatrix}\begin{bmatrix}7\\25\end{bmatrix}\begin{bmatrix}27\\18\end{bmatrix}$$
$$\begin{bmatrix}12\\22\end{bmatrix}\begin{bmatrix}18\\0\end{bmatrix}\begin{bmatrix}16\\7\end{bmatrix}\begin{bmatrix}17\\25\end{bmatrix}\begin{bmatrix}25\\25\end{bmatrix}$$

into the following set of 2×1 ciphertext vectors:

$$\begin{bmatrix}6\\4\end{bmatrix}\begin{bmatrix}9\\18\end{bmatrix}\begin{bmatrix}18\\19\end{bmatrix}\begin{bmatrix}2\\7\end{bmatrix}\begin{bmatrix}11\\24\end{bmatrix}\begin{bmatrix}19\\5\end{bmatrix}\begin{bmatrix}4\\27\end{bmatrix}\begin{bmatrix}28\\8\end{bmatrix}\begin{bmatrix}19\\21\end{bmatrix}\begin{bmatrix}22\\28\end{bmatrix}$$
$$\begin{bmatrix}25\\12\end{bmatrix}\begin{bmatrix}10\\22\end{bmatrix}\begin{bmatrix}24\\26\end{bmatrix}\begin{bmatrix}13\\18\end{bmatrix}\begin{bmatrix}9\\23\end{bmatrix}\begin{bmatrix}5\\19\end{bmatrix}$$

resulting in the encrypted message *b,li iwot cfwe, mkv wxnk _xnk _dun fayi ljew*. For readability, ciphertext blank spaces above are represented as "_".

2.4. Decryption Using the Algorithm

As with an Enigma Machine, decryption depends on knowing several specific parameters used for encryption including the character set, rotor matrices settings, and plugboard vector settings. Given the state of the rotors $(i, \sigma_3, \sigma_2, \sigma_1, \sigma_0)$, and using Tables 2 and 3, the matrix orientation settings can be set, and the appropriate inverse matrices used for decryption.

To decrypt the first two ciphertext characters *b,* ("b *comma*"), we first use Table 1 to write the ciphertext characters as an ordered pair (6, 4), and write the ordered pair as a 2×1 column vector

$$\begin{bmatrix}6\\4\end{bmatrix}.$$

We know these two ciphertext characters are the first two ciphertext characters in the ciphertext message, and therefore we know from the state diagram that $i = 0$, and that the ciphertext pair was encrypted with all rotor matrices in the initial position.

We can now proceed by multiplying the ciphertext column vector by the modular inverse of rotor matrix 3:

$$\begin{bmatrix}15 & 15\\23 & 2\end{bmatrix}\begin{bmatrix}6\\4\end{bmatrix} = \begin{bmatrix}5\\1\end{bmatrix},$$

next, multiply this resulting column vector by the modular inverse of rotor matrix 2:

$$\begin{bmatrix} 1 & 10 \\ 13 & 10 \end{bmatrix} \begin{bmatrix} 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 15 \\ 17 \end{bmatrix},$$

again multiply this resulting column vector by the modular inverse of rotor matrix 1:

$$\begin{bmatrix} 6 & 19 \\ 22 & 13 \end{bmatrix} \begin{bmatrix} 15 \\ 17 \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \end{bmatrix},$$

and finally multiply the resulting column vector above by the modular inverse of rotor matrix 0:

$$\begin{bmatrix} 4 & 27 \\ 5 & 18 \end{bmatrix} \begin{bmatrix} 7 \\ 0 \end{bmatrix} = \begin{bmatrix} 28 \\ 6 \end{bmatrix},$$

and now subtract mod 29 the plugboard column vector

$$\begin{bmatrix} 28 \\ 6 \end{bmatrix} - \begin{bmatrix} 20^0 \\ 12^0 \end{bmatrix} = \begin{bmatrix} 27 \\ 5 \end{bmatrix}.$$

The resulting column vector corresponds to the ordered pair (27, 5), which corresponds to the two plaintext characters *me*.

To decrypt the next pair, we note that now $i = 1$ and the state of rotor matrix 0 is now in orientation position 1, while the other three rotor matrices are still in orientation position 0, and we proceed as above to decrypt the next character pair. We continue this process until the entire message has been decrypted.For one more example, let uss take a random ciphertext pair and decrypt it. Consider the 11th pair "_d" (*blank space d*). The 11th pair corresponds to $i = 10$ (counting from $i = 0$). The corresponding ciphertext vector is

$$\begin{bmatrix} 25 \\ 12 \end{bmatrix}$$

and the state diagram for the 11th pair is (10, 0, 0, 2, 2) indicating $i = 10$ with rotor matrix 0 in state 2, rotor matrix 1 in state 2, and rotor matrices 2 and 3 both in state 0. Using the appropriately corresponding modular inverse rotor matrices from Table 3 above, we have the following computations: Multiply the ciphertext column vector by the modular inverse of rotor matrix 3 in state 0:

$$\begin{bmatrix} 15 & 15 \\ 23 & 2 \end{bmatrix} \begin{bmatrix} 25 \\ 12 \end{bmatrix} = \begin{bmatrix} 4 \\ 19 \end{bmatrix}.$$

Multiply the result by the modular inverse of rotor matrix 2 in state 0:

$$\begin{bmatrix} 1 & 10 \\ 13 & 2 \end{bmatrix} \begin{bmatrix} 4 \\ 19 \end{bmatrix} = \begin{bmatrix} 20 \\ 10 \end{bmatrix}.$$

Multiply the result by the modular inverse of rotor matrix 1 in state 2:

$$\begin{bmatrix} 13 & 22 \\ 19 & 6 \end{bmatrix} \begin{bmatrix} 20 \\ 10 \end{bmatrix} = \begin{bmatrix} 16 \\ 5 \end{bmatrix}$$

Multiply the result by the modular inverse of rotor matrix 0 in state 2:

$$\begin{bmatrix} 18 & 5 \\ 27 & 4 \end{bmatrix} \begin{bmatrix} 16 \\ 5 \end{bmatrix} = \begin{bmatrix} 23 \\ 17 \end{bmatrix}.$$

Subtract the result by the 10th plugboard vector:

$$\begin{bmatrix} 23 \\ 17 \end{bmatrix} - \begin{bmatrix} 20^{10} \\ 12^{10} \end{bmatrix} = \begin{bmatrix} 27 \\ 18 \end{bmatrix},$$

which corresponds to the plaintext characters *mi*. Continuing the decryption process for all vector pairs, then using the character – numerical assignment map above will show the original plaintext.

3. NOTES TO INSTRUCTOR

It is our hope that instructors may be able to modify this algorithm and the learning objectives below to introduce and enhance typical topics in a linear algebra course. The following learning objectives might be considered or modified for classroom use.

Learning Objective 1: Students will be able to explain historically accurate details from credible, authoritative sources about the World War II efforts to decrypt Enigma Machine ciphertext messages.

Note 1: The above questions allow students the opportunity to see the historical relevance of particular aspects of linear algebra and cryptography which may be used to spark a broad interest in possibly new areas of study. What countries were involved in the effort to decrypt Enigma encoded ciphertexts? How was the Enigma broken? Briefly explain what you learned about the story of breaking the Enigma in World War II.

Learning Objective 2: Students will be able to correctly calculate the order of the key space for invertible 2×2, 3×3, and 4×4 matrices using Rotman's result as stated in Overby et al. [20], given an appropriate prime modulus p by using hand calculation [20, p. 3]:

$$|GL(n, \mathbb{Z}_p)| = \prod_{i=0}^{n-1} (p^n - p^i)$$

Note 2: This objective gives students the opportunity both to see a technical result with possibly new notation and to consider the implication of the order of the key space with respect to the order of matrices used for rotors.

Learning Objective 3: Students will generate and explain two possible alternatives to the single column vector plugboard placement used in the algorithm.

Note 3: An interesting puzzle is to determine a plugboard vector that allows a single character to pass through the plugboard vector unchanged analogous to an Enigma plugboard, and to consider the implications for this possibility with respect to diffusion of plaintext characters in the encryption process.

Learning Objective 4: Students will choose a prime p for their modulus, determine a character-numerical map based on the prime chosen, and give an example plaintext message allowable with that character-numerical map based on the chosen modulus p.

Note 4: The implications of using a prime modulus is relevant to the order of the key space and impacts the character map set.

Learning Objective 5: Students will be able to explain a specific connection of the following components of the algorithm to other courses in mathematics and/or computer science, depending on background.

Note 5: In a spirit similar to Learning Objective 1 above, giving students the opportunity to connect to other courses they may have had in other disciplines allows them to see a bigger picture of the mathematics landscape than a microscopic view of plug-and-chug homework problems.

4. CONCLUDING REMARKS

We have presented a variation of the Hill Cipher that was modeled after a typical German Enigma Machine. The popularity and historical significance of German Enigma Machines, along with research over the past 20 years on variations of the Hill Cipher, present a unique opportunity to relate these two cryptosystems to each other as a vehicle to introduce students to important topics across the curriculum in mathematics and computing sciences. Using a series of 2×2 invertible matrices mod 29 as the rotors, and a 2×1 vector as a plugboard, and using a state data structure to track rotor matrix orientation, we mimic the electro-mechanical operation of a German Enigma Machine in a novel context of the Hill cryptosystem. By opening the matrix key space to all invertible matrices (mod 29) rather than using only involutory matrices, a larger matrix key space is possible. Some in-class discussion points were given that can be used to engage students while exploring aspects of mathematics and computing in the context of cryptography and historical contexts of the Hill Cipher and German Enigma Machines.

5. NOTES FOR FURTHER READING

For more details about German Enigma Machines, see [2, 3, 5, 7, 12, 14, 22, 23, 29].

For more details about the Hill Cipher, see [1, 4, 6, 9–11, 13, 15, 16, 17, 18, 19, 26, 28, 30].

ACKNOWLEDGMENTS

The authors wish to thank the reviewers for their thoughtful comments, suggestions for improvement, and encouragement.

ORCID

Porter E. Coggins III http://orcid.org/0000-0003-3734-5478
Tim Glatzer http://orcid.org/0000-0002-7057-4291

REFERENCES

1. Acharya, B., G. S. Rath, S. K. Patra, and S. K. Panigrahy. 2007. Novel method of generating self-invertible matrix for Hill Cipher Algorithm. *International Journal of Security*. 1(1): 14–21.
2. Cass, S. 2015. A simple enigma. *IEEE SPECTRUM*. 52(1): 19–20.
3. Dade, L. 2006. Enigma machine emulator. http://enigma.louisedade.co.uk/howitworks.html. Accessed 16 June 2017.
4. Farmanbar, M. and A. G. Chefranov. 2012. Investigation of Hill cipher modifications based on permutation and iteration. *International Journal of Computer Science and Network Security*. 10(9): 1–7.
5. Gillogly, J. J. 1995. Ciphertext-only cryptanalysis of Enigma. *Cryptologia*. 19(4): 405–413.
7. Hamamreh, R. A. and Farajallah, M. 2009. Design of a robust cryptosystem algorithm for non-invertible matrices based on Hill cipher. *International Journal of Computer Science and Network Security*. 9(5): 11–16.
6. Hamer, D. H. 1997. Enigma: Actions involved in the 'double stepping' of the middle rotor. *Cryptologia*. 21(1): 47–50.

8. Herstein, I. N. and D. J. Winter. 1988. *Matrix Theory and Linear Algebra*. New York, NY: Macmillan Publishing Company.
9. Hill, L. S. 1929. Cryptography in an algebraic alphabet. *American Mathematics Monthly*. 36(6): 306–312.
10. Hill, L. S. 1931. Concerning certain linear transformation apparatus of cryptography. *American Mathematics Monthly*. 38(3): 135–154.
11. Ismail, I. A., M. Amin, and H. Diab. 2006. How to repair the Hill cipher. *Journal of Zhejiang University-SCIENCE A*. 7(12): 2022–2030.
12. Kruh, L. and C. Deavours. 2002. The commercial Enigma: Beginnings of machine cryptography. *Cryptologia*. 26(1): 1–16.
13. Levine, J. and H. M. Nahikian. 1962. On the construction of involutory matrices. *American Mathematics Monthly*. 69(4): 267–272.
14. Lyons, J. 2009a. Enigma cipher. http://practicalcryptography.com/ciphers/enigma-cipher/. Accessed 16 June 2017.
15. Lyons, J. 2009b. Hill cipher. http://practicalcryptography.com/ciphers/hill-cipher/. Accessed 16 June 2017.
16. Lyons, J. 2009c. Practical cryptography, cryptanalysis of the Hill cipher. http://www.practicalcryptography.com/cryptanalysis/stochastic-searching/cryptanalysis-hill-cipher/. Accessed 16 June 2017.
17. Lyons, J. 2009d. Practical cryptography, cryptanalysis: Letter frequencies for various Languages. http://www.practicalcryptography.com/cryptanalysis/letter-frequencies-various-languages/. Accessed 16 June 2017.
18. McAndrew, A. 2009. Using the Hill cipher to teach cryptographic principles. *International Journal of Mathematical Education in Science and Technology*. 39: 967–979.
19. Nordin, M., M. N. A. Rahman, A. F.A. Abidin, M. K. Yusof, and N. S. M. Usop. 2013.Cryptography: A new approach of classical Hill cipher. *International Journal of Securityand its Applications*. 7(2): 179–190.
20. Overbey, J., W. Traves, and J. Wojdylo. 2005. On the keyspace of the Hill cipher. *Cryptologia*. 29(1): 59–72.
21. Ratcliff, R. A. 2003. How statistics led the Germans to believe Enigma secure and why they were wrong: Neglecting the practical mathematics of cipher machines. *Cryptologia*. 27(2): 119–131.
22. Rejewski, M. 1982. (Transl. C. Casparek). Mathematical solutions to the Enigma cipher. *Cryptologia*. 6(1): 1–18.
23. Rijmenants, D. 2004. Enigma message procedures. http://users.telenet.be/d.rijmenants/en/enigmaproc.htm. Accessed 16 June 2017.
24. Rijmenants, D. 2010. Enigma message procedures used by the Heer, Luftwaffe, and Kriegsmarine. *Cryptologia*. 34(4): 329–339.
25. Rosen, K. 2004. *Elementary Number Theory*, *Fifth Edition*, pp. 178–179. Reading, MA: Addison Wesley.
26. Sastry, V. U., N. R. Shankar, and S. D. Bhavani. 2010. A modified Hill cipher involving interweaving and iteration. *International Journal of Network Security*. 11(1): 11–16.
27. Shannon, C. 1949. Communication theory of secrecy systems. *Bell Systems Technical Journal*. 28(4): 656–715.

28. Stinson, D. R. 1995. *Cryptography: Theory and Practice*, p. 10. Boca Raton, FL: CRC Press.
29. Sullivan, G. and F. Weierud. 2005. Breaking German army ciphers. *Cryptologia*. 29(3): 193–232.
30. Toorani, M. and A. Falahati. 2009. A secure variant of the Hill cipher. In S. R. Thue, Y. Yang, and E. K. Park (Eds), *Proceedings of the 14th IEEE Symposium on Computers and Communications*, pp. 313–316. Piscataway, NJ: IEEE Press. 1530–1346.

BIOGRAPHICAL SKETCHES

Porter E. Coggins III has an undergraduate degree in mathematics and has previously written on the connection between zombie attacks and computer password security. He spends a lot of time explaining how to ride a bicycle when it is minus 30° F in Northern Minnesota. He also spends time looking for shortcuts in the basic algorithms for the 3×3 Rubik's cube, trying to explain to his son's three indoor cats that we can't put in a goldfish tank or let frogs and lizards run free indoors, and he excels at studying many languages both ancient and present, and fails in abundance in writing and speaking any of them well enough to order food or find the exit from either an ancient Egyptian pyramid or an ancient Greek temple.

After earning his doctoral degree in mathematics in West Virginia, Tim Glatzer headed out to the Rocky Mountain west, currently residing near Yellowstone National Park. He is a Project ACCCESS fellow and is a faculty advisor for the Northwest College chapter of Phi Theta Kappa. He is currently owned by a dog and a chinchilla, neither of whom seem terribly interested in his mathematical or educational endeavors. Unlike the first author, he stores his bike in the garage after the temperatures drop below 45 Fahrenheit.