Mathematics and Computer Science
2024, Vol. 9, No. 3, pp. 46-56
https://doi.org/10.11648/j.mcs.20240903.11

Research/Technical Note

Two Novel Multidimensional Affine Variations of the Hill Cipher

Porter Eldridge Coggins*

Department of Professional Education, Bemidji State University, Bemidji, United States of America

Abstract

Two novel symmetric multidimensional affine nested variations of the Hill Cipher are presented. The Hill Cipher is a block polygraphic substitution encryption scheme based on a linear transformation of plaintext characters into ciphertext characters. In the time since Hill first published his encryption scheme, variations, modifications, and improvements of theoretical and practical importance have been published every year indicating that the Hill Cipher is an active area of cryptography research. The first variation presented in this paper incorporated invertible key matrices of orders 2, 4, and 8 such that the matrix values of the *2×2* matrix rotate positions with each block of characters in a similar manner to the rotating letter wheels of a German Enigma Encoder, then results of the *2×2* key matrices output are passed to *4×4* key matrices, and *8x8* key matrix, *4×4* key matrices, and rotative-value *2×2* key matrices. The second variation is configured with invertible key matrices of orders 4, 8, and 16 without rotation of matrix values in a similar manner to the first variation. In both variations, plaintext characters of each block are operated on by exclusive-or (XOR) vectors prior to multiplication with the matrices to create the affine ciphers. Strengths, weaknesses, and other considerations are provided in the discussion. Two proposals are also argued with rationale for a more robust character set for encryption and the increase in modulus that the character set allows, and the possible advantages and disadvantages of affine XOR vectors.

Keywords

Cryptography, Hill Cipher, Matrix Theory, Invertible Matrices

1. Introduction

In 1929 [1] and 1931 [2] Lester Hill defined a symmetric encryption scheme based on a linear transformation of plaintext characters into ciphertext characters. In Hill's encryption algorithm, individual plaintext letters were assigned to corresponding numbers, then scrambled by matrix operations to create an encrypted *polygraphic* substitution ciphertext where blocks of plaintext characters were encrypted. The important feature that allowed decryption in Hill's Cipher was that the encryption key matrices were invertible. A representation of Hill's scheme can be seen in Figure 1 in which the matrix M_{3j} (:Matrix of order 3, j^{th} block) is subscripted with the number 3 representing an invertible and block number *j*, and specifically, an involutory *3x3* matrix. In Hill's scheme, a plaintext message is broken into *j* blocks where the block size is dependent upon the dimension *n*, for a square *nxn* invertible matrix. He also represented his affine scheme via a system of linear equations where x_i and y_i represent the plaintext and ciphertext respectively and a_i is the affine element compo-

*Corresponding author: porter.coggins@bemidjistate.edu (Porter Eldridge Coggins)

Received: 14 June 2024; **Accepted:** 2 July 2024; **Published:** 23 July 2024

Copyright: © The Author(s), 2024. Published by Science Publishing Group. This is an **Open Access** article, distributed under the terms of the Creative Commons Attribution 4.0 License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution and reproduction in any medium, provided the original work is properly cited.

nent as:

$y_{ij} = a_{ij}x_i + a_i$ over a field.

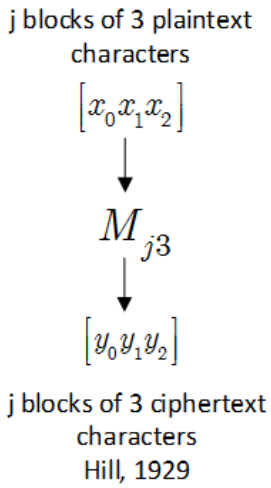


Figure 1. *Hill's Encryption Scheme transforming j blocks at a time via an invertible 3x3 matrix.*

Variations and application of the Hill Cipher continue to generate significant research as of 2024. Over the time span between Hill's original publication in 1929 through 2024 [3], researchers have proposed applications to elliptic curve cryptography [4, 5], RSA cryptography [6], image security [5, 7, 8], public key cryptography [9], methods of generating key matrices and various types of key matrices (involutory, circulant, non-involutory-non-circulant) [10-13], connection to German Enigma Encoders [14], and Rivest Codes [15], among many other topics [16-20]. However, the literature is silent on two important ideas. First, although models of using double and triple Hill Cipher encryptions have been published [19] along with well-known issues of the relationship between multiple encryptions of the same key with respect to security, to date, no one has presented a scheme for a multidimensional version of the Hill Cipher in which key matrices of different orders are aligned in the encryption sequence. Second, and interestingly, most examples, whether in character or binary formats, typically use some version of Hill's original character-map scheme of assigning the 26 characters in the English Language. The consequences of this limited character set has direct implications regarding the key space for matrices used. Both of these omissions and gaps in the literature will be explained below.

2. The First Variation

First, the key space requirements are presented in Table 1. Second, the scheme of the first variation is represented in Figure 2. Finally, a stepwise procedure of the algorithm is presented corresponding to the number steps along the left side of Figure 2.

2.1. Key Space of First Variation

Table 1 represents the number of pseudo-key XOR (XOR: exclusive or function) row vectors and invertible key matrices for the first variation. The pseudo-key XOR row vectors serve as operating on the initial plaintext character blocks and operating on the outputs of each stage prior to input to the next stage of the algorithm. The danger of unwisely choosing these pseudo-key XOR row vectors will be stated in section 4.2. These vectors are not keys in the traditional sense, hence are called *pseudo-keys*.

Table 1. *Key Space Requirements for First Variation.*

Object	Dimension	Number Required
Pseudo-Key XOR Row Vector	1x2	12
Pseudo-Key XOR Row Vector	1x4	2
Pseudo-Key XOR Row Vector	1x8	3
Key Matrix	2x2	8
Key Matrix	4x4	4
Key Matrix	8x8	1

* Number Required for 2-4-8 Dimension Hill Variation.

2.2. Encryption Scheme of First Variation

Reference to the left-hand numbers in Figure 2 will be made throughout the explanation of the first variation below. One minor notational device should be mentioned: without loss of generality and with a minor abuse of notation, the forward slashed down arrow $\not\downarrow$ symbol normally represents an input (down-arrow) and the number of bits (to the right of the forward slash /). However, the Hill Cipher literature traditionally represents alphabet characters as both input and output, rather than a number of bits. Following this practice, and in slight abuse of the bit-use in conventional encryption-decryption diagrams, the number represented in Figure 2 in the upper right of the forward slash / will represent the *number of alphabet characters* that will input in the direction of the down-arrows rather than the number of bits.

Researchers have variously used as keys: powers of a single key matrix of the same dimension [20], *semi cipher block chaining* [17, 18] using a string of 3 matrix keys of the same dimension with modified XOR inputs, multiple keys including a case 24 key matrices of the same dimension with linear feedback shift register operations [19], and a string of rotating matrices of the same dimension that was somewhat similar to the German Enigma Encoder (Figures 3 and 4) [14].

The first variation presented in Figure 2 differs from all previously known variations due to the use of multidimen-

sional matrices, that is, matrices of *different* orders. In this first variation, a combination of *2x2*, *4x4*, and *8x8* matrices are used. Refer to Figure 2 and the step numbers in the left column of that figure.

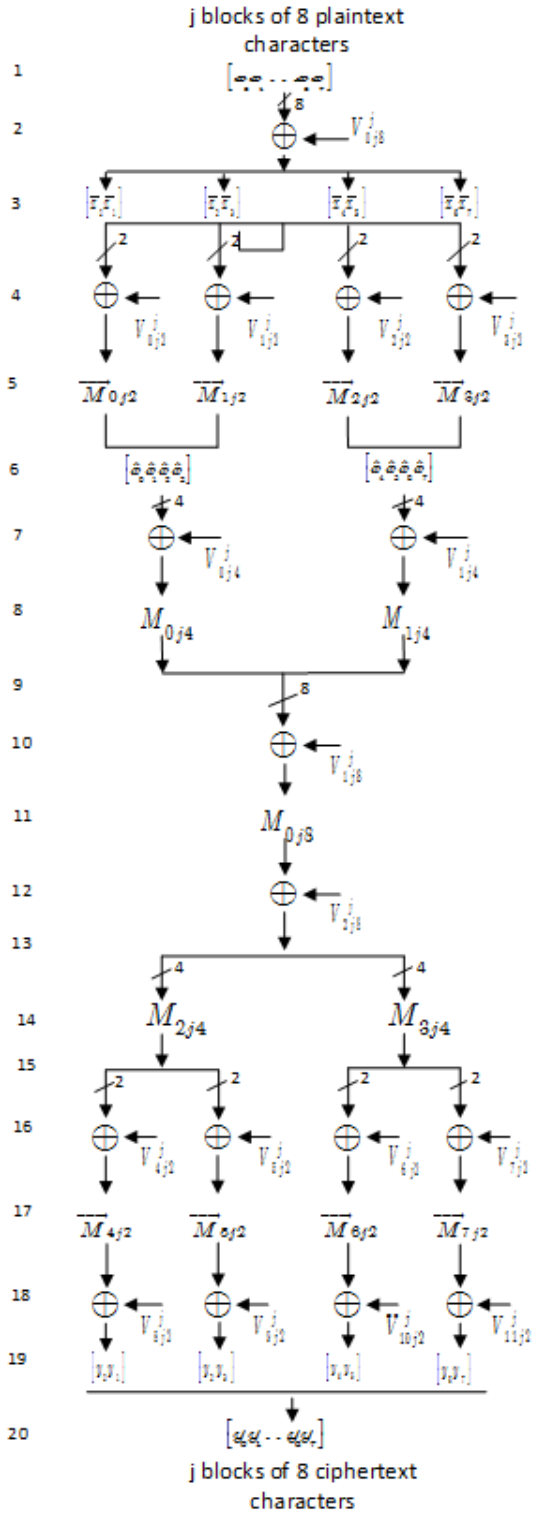


Figure 2. *Scheme of First Variation.*

Step 1: *1x8* block of 8 plaintext characters entered as a row vector $\left[x_{0j}x_{1j}\ldots x_{6j}x_{7j}\right]$ for characters x_{ij} (x_{ij} : i^{th} element of the j^{th} block) where i is the character number and j is the block.

Pad the plaintext if necessary to ensure the plaintext is a multiple of 8 characters. Padding can be determined using any of several possibilities including an agreed upon padding string that will not affect the semantics of the plaintext when decrypted, a set of printable control characters, or another protocol agreed upon prior to encryption.

The reference value j follows each block through the algorithm and sometimes may also serve as an exponent modulo prime p for the pseudo-key xor row vectors. All vector characters are subscripted 0 through 7 along with the block j and dimension.

Step 2: XOR the first random *1x8* pseudo-key row vector raised to the j^{th} power with the j^{th} block of *1x8* row vector characters: $\left[x_{0j}x_{1j}x_{2j}\ldots x_{6j}x_{7j}\right]$ XOR $\left[v_{0j}v_{1j}v_{2j}\ldots v_{6j}v_{7j}\right]^j$ modulo p, where $\left[v_{0j}v_{1j}v_{2j}\ldots v_{6j}v_{7j}\right]^j$ is interpreted as $\left[v^j_{ij} \bmod p\right]$ for i = 0 to 7.

1) The pseudo-key xor row vectors are subscripted as follows:
   a. assigned vector number to keep track of each vector.
   b. the block number j.
   c. the dimension of the xor pseudo-key row vector as appropriate (either 2, 4, or 8).
2) Raising each pseudo-key xor vector to the j^{th} power *potentially* aids in confusion [20].
3) Pseudo-key xor vectors may enhance the ciphertext character distribution up to probability when the plaintext character distribution is considered.
4) Raising each pseudo-key xor vector the j^{th} power (modulo prime p) allows each pseudo-key xor vector to change vector values (up to modulo prime p before repeating the original values) and adds variability into each pseudo-key xor vector.

Step 3: Break the output of step 2 into 4 pairs of *1x2* row vectors (denoted as x-bar, $\left[\bar{x}_i\bar{x}_{i+1}\right]$, in Figure 2) to xor in step 4.

Step 4: XOR individually each of the first four *1x2* output vectors $\left[\bar{x}_i\bar{x}_{i+1}\right]$ in the above step with the j^{th} power each of one of the 4 random *1x2* pseudo-key xor row vectors respectively.

1) XOR pairs $\left[\bar{x}_{ij} \oplus \left(v^j_{ij}\right) \bmod p\right]$ for i = 0, 3 in Figure 2.
2) Comment in Step 2 a) and b) apply here regarding subscripts and cryptographic confusion [21] (when properly constructed).

Step 5: Multiply each of the output vector pairs from Step 4 above with one of 4 random invertible *2x2* matrices respectively.

1) All matrices are invertible but neither involutory (self-invertible) nor circulant. Checking that the matrices follow all of the conditions above is a relatively

simple process by testing whether an invertible matrix $A = A^{-1} or A(A^{-1}) = I$ where I is the identity matrix, and by inspection that the matrix is not circulant.

2) Matrices are subscripted as follows-
   a. Assigned matrix number *i*.
   b. The block number *j*.
   c. The dimension of the matrix appropriate to the input vector.
3) With each new block *j* of plaintext, one *2x2* matrices rotate clockwise the values of the matrix when $j \equiv i \bmod 8$, Table 2, in a similar manner to rotor wheels in German Enigma Encoders shown in Figures 3 and 4.
4) Matrices rotate on a schedule with respect to the assigned matrix number as follows following an established matrix-value rotation protocol [18]:

When the block number *j* is congruent to the assigned matrix number *i* modulo 8, rotate clockwise one-quarter each of the matrix values of matrix *i*.

5) State *S* is defined to be the matrix configuration based on the number of blocks in the plaintext (with padding to mod 8).
6) The length of the plaintext, *len*, (*len*: number of plaintext characters) is essentially the total number of characters in the plaintext/ciphertext. The number *k* of blocks *j* of 8 characters is $k = (len \bmod 8) + (8 - (len \bmod 8))$ using appropriate padding.
7) The state of each *2x2* rotor matrix in the encryption process can be determined in advance of any encryption with the numbers *k* and *j* which are both based on the plaintext length. Both the length of the plaintext and the state ***S*** of all 8 matrices of order 2 with respect to the j^{th} block will allow tracking necessary for decryption where the block *j*=0 is the initial state of the matrices with respect to the original orientation of their matrix values. Since there is a one-to-one correspondence between plaintext and ciphertext characters, the length can be determined, and from the length, the state of all order 2 matrices can be calculated in a similar manner to an established protocol [17]. See Table 2 for the 2x2 matrix configurations with respect to the plaintext character length and block *j*.

Step 6: Combine the 4 output *1x2* vectors of the step above into 2 input *1x4* vectors, $\left[\hat{x}_0 \hat{x}_1 \hat{x}_2 \hat{x}_3\right]$ and $\left[\hat{x}_4 \hat{x}_5 \hat{x}_6 \hat{x}_7\right]$.

Step 7: XOR each *1x4* vector respectively with a *1x4* pseudo-key vector:

$\left[\hat{x}_{ij} \oplus \left(v^j{}_{ij} \bmod p\right)\right]$ for i = 0, 1 in Figure 2.

Step 8: Multiply the output vector from Step 7 with one of two random invertible *4x4* matrices respectively.

Comments from Step 5 apply with appropriate change of dimension.

Step 9: Combine the output of the product of the two *1x4* vector and *4x4* matrices into a single *1x8* vector.

Step 10: XOR the output of Step 9 with the second random xor pseudo-key *1x8* vector.

This is essentially a repeat of Step 2 with appropriate vectors.

Step 11: Multiply the output of Step 10 with an *8x8* invertible matrix.

This invertible *8x8* matrix is neither involutory nor circulant.

Step 12: XOR the output of Step 9 with the second random xor pseudo-key *1x8* vector.

Step 13: Split the output *1x8* row vector from Step 12 into a lower half *1x4* row vector and an upper half *1x4* row vector.

Step 14: Multiply each 1x4 row vector independently by the respective third and fourth random invertible *4x4* matrices respectively.

These *4x4* matrices are neither involutory nor circulant.

Step 15: Split the output of each *1x4* row vector from Step 14 into a lower half *1x2* row vector and an upper half *1x2* row vector.

There will be four total *1x2* row vectors from Step 15, two each for each of the two 1x4 row vectors from Step 14.

Step 16: XOR each output of Step 15 with random xor pseudo-key *1x2* vectors respectively.

1) with the j^{th} power each of one of the 4 random *1x2* pseudo-key xor row vectors respectively.
2) XOR pairs $\left[\bar{\bar{x}}_{ij} \oplus \left(v^j{}_{ij} \bmod p\right)\right]$ for *i* = 4, 7 in Figure 2.

Step 17: Similar to Step 5, Multiply the row vector output of Step 16 with four new random invertible *2x2* matrices. Matrices rotate once clockwise with each block *j*:

1) Matrices rotate on a schedule with respect to the assigned matrix number following Steps 5.c) and 5.d):

When the block number *j* is congruent to the assigned matrix number *i* modulo 8, rotate clockwise one-quarter each of the matrix values of matrix *i*.

2) Both the length of the plaintext and the state ***S*** with respect to the j^{th} block will allow tracking necessary for decryption where the block *j*=0 is the initial state of the matrices with respect to the original orientation of their matrix values.

Step 18: XOR each output of Step 17 with 4 new random xor pseudo-key *1x2* vectors respectively.

XOR pairs $\left[\bar{\bar{\bar{x}}}_{ij} \oplus \left(v^j{}_{ij} \bmod p\right)\right]$ for *i* = 8, 11 in Figure 2.



Figure 3. *A Typical German Enigma Encoder with respect to Step 5. Used with permission from copyright owner Enigma Museum: http://EnigmaMuseum.com©.*

Step 19: The four resulting *1x2* row vectors from Step 18 constitute the ciphertext in four subblocks.

Step 20: Final step, concatenate the four *1x2* row vectors from Step 19 into a single *1x8* row vector $\left[y_{0j}y_{1j}\ldots y_{6j}y_{7j}\right]$ representing the j^{th} block of plaintext $\left[x_{0j}x_{1j}\ldots x_{6j}x_{7j}\right]$.

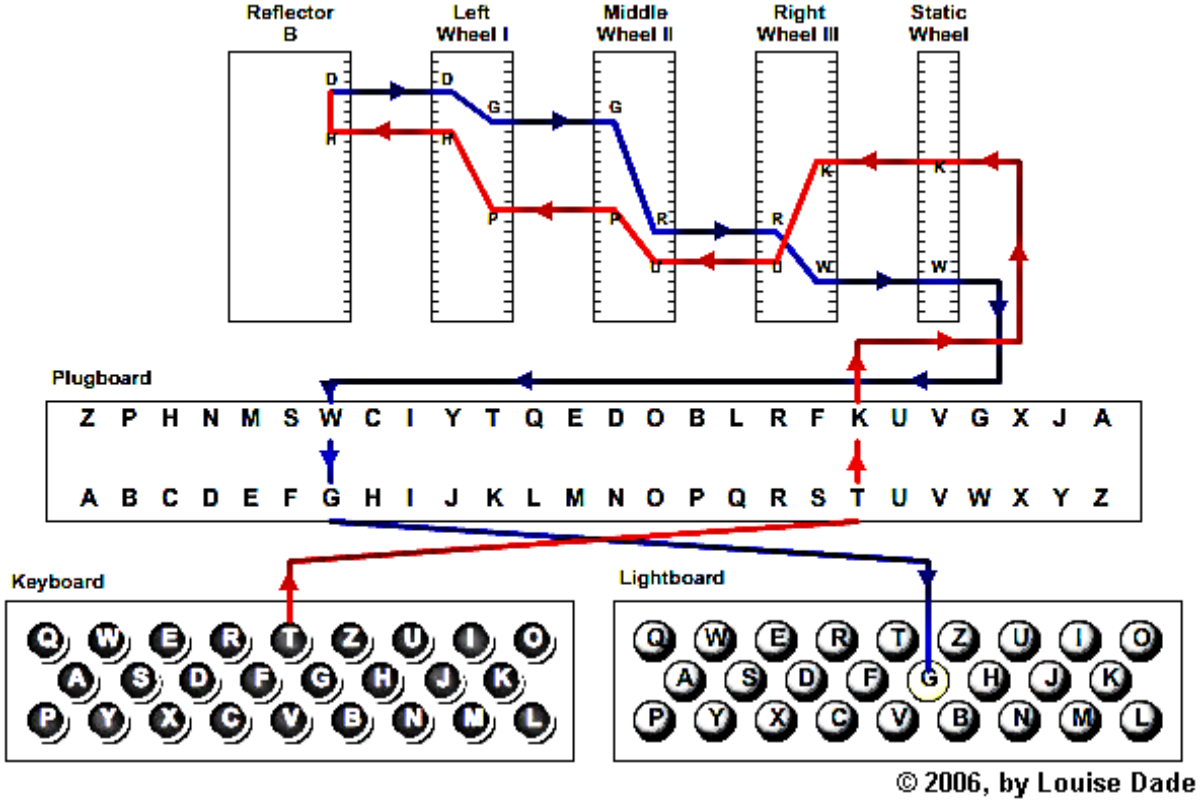


Figure 4. *Signal Flow of a German Enigma Encoder with respect to Step 5. Used with permission from copyright owner Louise Dade©.*

2.3. State Tracking of *2x2* Rotor Matrices

It is necessary to keep track of the state *S* of each *2x2* rotor matrix in order to determine how to orient the rotor matrices for decryption. We will use the following convention as an initial rotor matrix position with respect to the normal fixed matrix element *subscript* positions of a matrix element a_{ij} for *i,j* = 1 to 2: $\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix}$. Let the matrix value *a* in the *2x2* rotor matrices be rotated as clockwise one-quarter turn in Step 5 with respect to Figure 2 as further explained below. For ease of visualization, the matrix values are noted in alphabetical order. The convention will be to rotate the values of the *2x2* matrices for each block *j* $\begin{bmatrix} a & b \\ d & c \end{bmatrix} \rightarrow \begin{bmatrix} d & a \\ c & b \end{bmatrix} \rightarrow \begin{bmatrix} c & d \\ b & a \end{bmatrix} \rightarrow \begin{bmatrix} b & c \\ a & d \end{bmatrix}$. The initial value positions will be the assignment $a \rightarrow x_{11}, b \rightarrow x_{12}, c \rightarrow x_{21}, d \rightarrow x_{22}$. The state can now be stated as in Table 2 with respect to the plaintext length, assigned matrix number *i*, and the j^{th} block. For example, with a plaintext character length of 8, and the 0^{th} block, the matrix values are in the initial state. Matrix values will rotate clockwise one-quarter turn such that the matrix values will now be $a \rightarrow x_{12}, b \rightarrow x_{22}, c \rightarrow x_{21}, d \rightarrow x_{11}$ as $\begin{bmatrix} d & a \\ c & b \end{bmatrix}$. Rotate clockwise one-quarter turn the matrix M_{i2j} (M_{i2j}: i^{th} *2x2* Matrix, j^{th} block) when $j \equiv i \bmod 8$. Initialize $M_{020} \rightarrow \begin{bmatrix} b & c \\ a & d \end{bmatrix}$ prior to passing the first block *j = 0* through the encryption process. Once the plaintext character length is known, which also means the ciphertext character length is known, the number of 8-character blocks *j* is known. By the rotation scheme above, it is deterministic to configure each of the eight *2x2* matrices in the correct orientation for decryption. Hence, the block number corresponds to which *2x2* matrices have rotated and taking the block number mod 8 indicates how many times the particular *2x2* matrix has rotated, which indicates which state the particular *2x2* matrix is in according to Table 2. The integer portion of $\frac{j}{8}$ represents the number of times all matrices have rotated. When the block

number j is a multiple of 8, all of the *2x2* matrices are in State 0. State configuration for *2x2* rotor Matrix M_{i2j} is $j \equiv i \bmod 8$.

Table 2. *State of 2x2 Rotor Matrices.*

State 0	State 1	State 2	State 3
$\begin{bmatrix} a & b \\ d & c \end{bmatrix}$	$\begin{bmatrix} d & a \\ c & b \end{bmatrix}$	$\begin{bmatrix} c & d \\ b & a \end{bmatrix}$	$\begin{bmatrix} b & c \\ a & d \end{bmatrix}$

State configurations for each 2x2 rotor Matrix M_{i2j} for $j \equiv i \bmod 8$

2.4. Decryption of First Variation

Decryption is possible with an additional preliminary Step 0, not shown on Figure 2. This preliminary step is accomplished by taking the ciphertext character count *i*, determining the number of blocks *j*, and setting the state of each *2x2* matrix according to the state rotation schedule in Table 2.

Due to the ability to *unwind* the ciphertext because all matrices and all XOR operations are invertible. The state of each 2x2 matrix can be determined by the ciphertext length by section. 2.3 above.

Attention must be made to the state of the *2x2* matrices because these *2x2* matrix values will now rotate counterclockwise for decryption, for each block *j* of characters according to the rotation scheme of rotating the i^{th} *2x2* matrix for each block *j* when $j \equiv i \bmod 8$. The ciphertext length, *len*, is known which also determines the padding number. From that, the number of *k* is known, thus, the state of each *2x2* matrix, that is the number of times each matrix has been rotated one-quarter turn clockwise, can be determined. For each block *j* of ciphertext characters, each *2x2* matrix will be set to the appropriate state with respect to matrix-value positions and reverse the rotation direction to counterclockwise for each block *j*.

Once the states of all of the *2x2* matrices are set, decryption proceeds from the bottom up in Figure 2 ensuring to use the inverse of each matrix.

2.5. Remarks on First Variation

It is well known that using multiple key matrices of the same order does not necessarily improve security using the Hill Cipher. Therefore, the following points are made. First, in this first variation example, three matrices of different dimensions, orders 2, 4, and 8, are used for encryption. Second, XOR is repeatedly used as an affine operation, which can increase security if appropriately used. Third, although impractical on some level, the values of all 8 matrices of order 2 are rotated in different ways somewhat similar to the rotor wheels in a typical German Enigma Encoder. Fourth, this first variation increases confusion [21], possibly mitigating some of the Hill Cipher vulnerabilities of various attacks [17]. Further remarks will be stated below in the implications section 4.

3. Second Variation

First, the key space requirements are presented in Table 3. Second, the scheme of the first variation represented in Figure 5. Finally, a stepwise procedure of the algorithm is presented corresponding to the number steps along the left side of Figure 5.

3.1. Key Space of Second Variation

Table 3 represents the number of pseudo-key XOR row vectors and key matrices for the second variation. As in the first variation, the pseudo-key XOR row vectors serve as encryption functions operating on the initial plaintext character blocks and operating on the outputs of each stage prior to input to the next stage of the algorithm. The danger of unwisely choosing these pseudo-key XOR row vectors will be stated in the section 4.2.

Table 3. *Key Space Requirements for Second Variation.*

Object	Dimension	Number Required
Pseudo-Key XOR Row Vector	1x4	12
Pseudo-Key XOR Row Vector	1x8	2
Pseudo-Key XOR Row Vector	1x16	3
Key Matrix	4x4	8
Key Matrix	8x8	4
Key Matrix	8x8	1

* Number Required for 4-8-16 Dimension Hill Variation.

3.2. Encryption Scheme of Second Variation

Again, reference to the left-hand numbers in Figure 5 will be made throughout the explanation of the second variation below. Again, we make a minor abuse of notation regarding the forward slashed down arrow $\not\downarrow$ symbol. Normally the down-arrow represents an input and the number of bits (to the right of the forward slash /). And again, traditionally the Hill Cipher literature nearly always represents alphabet characters as both input and output, rather than a number of bits. Following this tradition, and in slight abuse of the bit-use in conventional encryption-decryption diagrams, the number

represented in Figure 5 in the upper right of the forward slash / will represent the *number of alphabet characters* that will input in the direction of the down-arrows rather than the number of bits.

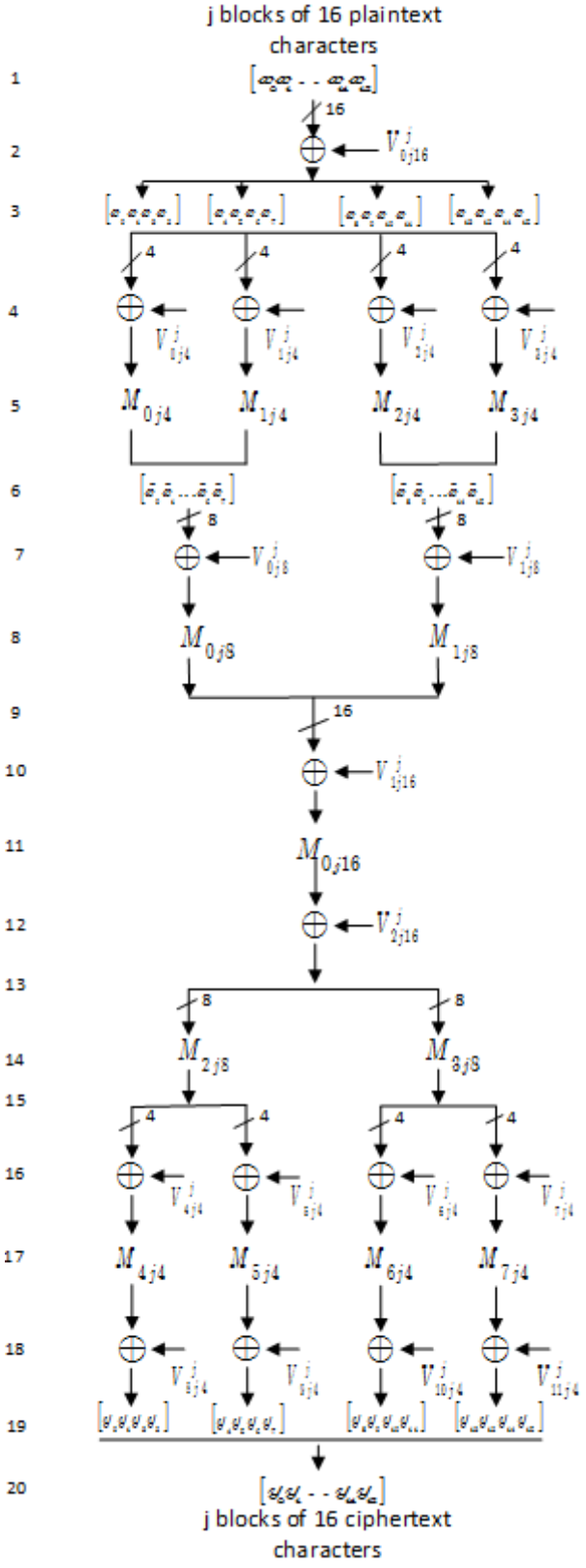


Figure 5. *Scheme of Second Variation.*

Step 1: *1x16* block of 16 plaintext characters entered as a row vector $\left[x_{0j}x_{1j}x_{2j}\ldots x_{14j}x_{15j}\right]$ for character x_{ij} where *i* is the character number and *j* is the block number.

The reference value *j* follows each block through the algorithm also serves as an exponent modulo prime *p* for the pseudo-key xor row vectors. All vector characters are subscripted 0 through 15.

Step 2: XOR the first 1x16 pseudo-key row vector block of characters raised to the j^{th} power of first random 1x16 row vector designated in the vector subscript. That is, $\left[x_{0j}x_{1j}x_{2j}\ldots x_{14j}x_{15j}\right]$ XOR $\left[v_{0j}v_{1j}v_{2j}\ldots v_{14j}v_{15j}\right]^j$ modulo *p*, where $\left[v_{0j}v_{1j}v_{2j}\ldots v_{14j}v_{15j}\right]^j$ is interpreted as $\left[v^j{}_{ij} \bmod p\right]$ for i = 0 to 15.

1) The pseudo-key xor row vectors are subscripted as follows:
   a. assigned vector number to keep track of each vector.
   b. the block number *j*.
   c. the dimension of the xor pseudo-key row vector as appropriate (either 4, 8, or 16).
2) Raising each pseudo-key xor vector to the j^{th} power *potentially* aids in confusion [21].
3) Pseudo-key xor vectors may enhance the ciphertext character distribution up to probability when the plaintext character distribution is considered.
4) Raising each pseudo-key xor vector the j^{th} power (modulo prime *p*) allows each pseudo-key xor vector to change vector values (up to modulo prime *p* before repeating the original values) and adds variability into each pseudo-key xor vector.

Step 3: Break the output of step 2 into 4 pairs of *1x4* row vectors denoted as x-bar, $\left[\overline{x}_i\overline{x}_{i+1}\overline{x}_{i+2}\overline{x}_{i+3}\right]$, in Figure 5 to xor in step 4.

Step 4: XOR individually each of the first four *1x4* output vectors $\left[\overline{x}_i\overline{x}_{i+1}\overline{x}_{i+2}\overline{x}_{i+3}\right]$ in the above step with the j^{th} power each of one of the 4 random *1x4* pseudo-key xor row vectors respectively.

1) XOR pairs $\left[x_{ij} \oplus \left(v^j{}_{ij} \bmod p\right)\right]$ for i = 0, 3 in Figure 5 remembering these computations are preformed modulo *p* for prime *p*.
2) Comment in Step 2 a) and b) apply here regarding subscripts and cryptography confusion [21] (when properly constructed).

Step 5: Multiply each of the output vector pairs from Step 4 above with one of 4 random invertible *4x4* matrices respectively.

1) All matrices are invertible but neither involutory (self-invertible) nor circulant. Checking that the matrices follow all of the conditions in a) is a relatively simple process by testing whether an invertible matrix $A = A^{-1} or A(A^{-1}) = I$ where *I* is the identity matrix, and by inspection that the matrix is not circulant.

2) Matrices are subscripted as follows-

a. Assigned matrix number.
b. The block number *j*.
c. The dimension of the matrix appropriate to the input vector.

Step 6: Combine the four output 1x4 vectors of the step above into two input *1x8* vectors, $\left[\hat{x}_0\hat{x}_1\ldots\hat{x}_6\hat{x}_7\right]$ and $\left[\hat{x}_8\hat{x}_9\ldots\hat{x}_{14}\hat{x}_{15}\right]$.

Step 7: XOR each *1x8* vector respectively with a *1x8* pseudo-key vector.

$\left[\tilde{x}_{ij}\oplus\left(v^j{}_{ij} \bmod p\right)\right]$ for i = 0, 1 in Figure 5 ensuring all computations are completed modulo *p* for prime *p*.

Step 8: Multiply the output vector from Step 7 with one of 2 random invertible *8x8* matrices respectively.

Comments from Step 5 apply with appropriate change of dimension.

Step 9: Combine the output of the product of each *1x8* vector and *8x8* matrices into a single *1x16* vector.

Step 10: XOR the output of Step 9 with the second random xor pseudo-key *1x16* vector.

This is essentially a repeat of Step 2 with appropriate vectors.

Step 11: Multiply the output of Step 10 with an *16x16* invertible matrix.

This invertible *16x16* matrix is neither involutory nor circulant.

Step 12: XOR the output of Step 9 with the second random xor pseudo-key *1x16* vector.

Step 13: Split the output *1x16* row vector from Step 12 into a lower half *1x8* row vector and an upper half *1x8* row vector.

Step 14: Multiply each *1x8* row vector independently by the respective third and fourth (numbered as matrix 2 and matrix 3) random invertible *8x8* matrices respectively.

These *8x8* matrices are neither involutory nor circulant.

Step 15: Split the output of each *1x8* row vector from Step 14 into a lower half *1x4* row vector and an upper half *1x4* row vector.

There will be four total *1x4* row vectors from Step 15, two each for each of the two *1x8* row vectors from Step 14.

Step 16: XOR each output of Step 15 with random xor pseudo-key *1x4* vectors respectively with the j^{th} power each of one of the 4 random *1x4* pseudo-key xor row vectors respectively.

XOR pairs $\left[\bar{\tilde{x}}_{ij}\oplus\left(v^j{}_{ij} \bmod p\right)\right]$ for i = 4, 7 in Figure 5.

Step 17: Similar to Step 5, Multiply the row vector output of Step 16 with four new random invertible *4x4* matrices.

Step 18: XOR each output of Step 17 with 4 new random xor pseudo-key *1x4* vectors respectively.

XOR pairs $\left[\bar{\bar{\tilde{x}}}_{ij}\oplus\left(v^j{}_{ij} \bmod p\right)\right]$ for i = 8, 11 in Figure 5.

Step 19: The four resulting *1x4* row vectors from Step 18 constitute the ciphertext in four subblocks.

Step 20: Final step, concatenate the four *1x4* row vectors from Step 19 into a single *1x16* row vector $\left[y_{0j}y_{1j}\cdots y_{14j}y_{15j}\right]$ representing the j^{th} block of plaintext $\left[x_{0j}x_{1j}\cdots x_{14j}x_{15j}\right]$.

3.3. Decryption of Second Variation

Decryption of the second variation (Figure 5) follows similarly to decryption of the first variation with the exception of the lack of rotor matrices in the second variation and no need of state tracking. All key matrices and the xor-operation are invertible. The ciphertext can be immediately operated on in a bottom-up fashion in Figure 5.

3.4. Remarks on Second Variation

This second variation used key matrices of orders 4, 8, and 16 that increase the confusion [21] between the plaintext and the encryption keys. Remarks in section 2.5 apply other than comments on matrix-value rotation of the *2x2* matrices. Additionally, this second variation fills a gap in the literature in which matrices, no matter how many are used in the encryption process, all use a matrix of order *n*. This second variation is also susceptible to known attacks [17], although these various attacks *may* take longer to decrypt due to the orders of the key matrices and xor-functions used.

4. Implications

4.1. Character Set Implications

The character map of character-number assignments plays a significant role in the Hill Cipher that has not been addressed in the literature. Both the size of the matrix and the size of the character set used play a significant role with respect to the total keyspace of invertible matrices.

Although not all permutation of matrix values lead to an invertible matrix, the number of permutations does give an upper bound to the possible number of invertible matrices. The left column in Table 4 represents the number of characters in the *language* of the plaintext character set. The center column of Table 4 represents the order of the matrix, *nxn*. The right column indicates the number of permutations the plaintext characters can be in. The number of characters in the set (the modulus) influences boundaries of the correspondence between character-numerical value assignments. Again, not all possible permutations yield an invertible matrix, rather the number of possible permutations in an upper bound on the total number of invertible matrices.

Table 4. *Modulus and Order of the Matrices.*

Modulus	nxn Matrix	Number of Possible Permutations in Matrix slots $\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}$
26	2x2 Matrix	26^4=456,976
29	2x2 Matrix	29^4=707,281
191	2x2 Matrix	191^4=1,330,863,361
191	4x4 Matrix	191^6>3.137E36
191	8x8 Matrix	191^{64}>9.685E145
191	16x16 Matrix	191^{256}>8.801E583

The formula for determining the total number of invertible *nxn* matrices (including involutory matrices and circulant matrices) is well known [22, 23]. Table 5 represents $GL\left(2,\mathbb{Z}_p\right)$: the General Linear Group of invertible *2x2* matrices over the integers $\mathbb{Z}$ modulo prime *p* [23] as an example illustrating for a constant order 2 matrix, the number of invertible matrices increases as the character set increases. Understanding that small-order matrices (2, 3, sometimes 4) are typically given in the literature when examples are explained and calculated, and typically for academic journals requiring the English Language, 26 is nearly always chosen as the modulus (26 characters in the English alphabet). Using ASCII ISO 8851-1 characters excluding unprintable control characters as the domain, the encryption character set is increased to 191 characters and creates a mapping that to the codomain which (ASCII ISO 8851-1). A function $e\left(x_{ij}\right)$ can be created which avoids the control characters with $e\left(x_{ij}\right)=\begin{cases} e_1\left(x_{ij}\right) \\ e_2\left(x_{ij}\right) \end{cases}$. Even though a message may only be defined to say the English alphabet, ISO 8851-1 allows greater semantic clarity and disambiguation.

Table 5. *Invertible 2x2 Matrices over various Moduli.*

Modulo p	$GL\left(2,\mathbb{Z}_p\right)=\prod_{i=0}^{2-1}\left(p^2-p^i\right)$
26	157,248
29	682,080
191	1,323,859,200

As the modulus increases, the number of invertible matrices with constant order increases

Tables 4 and 5 highlight the greater the number of elements in a character set, the greater the number of possible and actual invertible matrices, hence the greater the *confusion* [21] between the key matrices and the ciphertext.

4.2. XOR Pseudo-Key Vector Implications

Both of the multidimensional Hill Cipher variations presented above made use of xor functions between a *random* vector (raised to the j^{th} power) and components of the plaintext and output of key matrices. This either can increase or decrease diffusion [21]. That is, the intermediate text and ciphertext may either be closer to or further from a uniform distribution. Analysis of these types of vectors used in xor-functions will not be given here, it is worth careful consideration if the use of such xor-function vectors are used.

5. Conclusion and Recommendations

This paper described two multidimensional affine variations of the Hill Cipher as well as implications between the encryption character set and pseudo-key xor-function vectors. The novelties of the two variations were that each used several orders of key matrices (either orders 2, 4, and 8 or orders 4, 8, and 16) with the first variation rotating matrix values of each *2x2* matrix depending on the block *j*. These two variations using multidimensional affine encryption schemes have not previously been reported in the literature.

Recommendations for further study include: an analysis of each of the schemes using specific well-chosen pseudo-key xor-vectors, key matrices, and a significant length plaintext, the consequences of poorly chosen xor-functions, the determination of a rotation schedule that admits invertible matrices of orders greater than 2, and the use of genetic programming to determine variations of the numbers and orders of invertible key matrices and pseudo-key xor-vectors that meet specific encryption analytics that incorporate key matrix generation ideas from previous work [10-13]. Genetic algorithms have been used in the past [12], but the genetic programming approach would a significant advancement in key space of the Hill Cipher.

Abbreviations

M_{inj}	j^{th} Block of the i^{th} Matrix of order n
$V_{ij}^{\ j}$	j^{th} Block of the i^{th} Character Raised to the j^{th} Power
XOR	Exclusive-or Function
$GL\left(2,Z_p\right)$	General Linear Group of degree 2 (*2x2*) Invertible Matrices Over the Integers, $\mathbb{Z}$ of Prime *p*.

Supplementary Material

The supplementary material can be accessed at https://doi.org/10.11648/ j.mcs.20240903.11

Acknowledgments

I wish to thank Dr. Tom Perera at the Enigma Museum for granting permission to use the German Enigma Encoder photo. I wish to thank Louise Dade for granting permission to use her Enigma Emulator schematic diagram. I wish to thank Dr. Halbana Tarmizi for introducing me to Visio to produce the vector figures used in this paper. I wish to thank both Dr. Lakshika Gunawardana and Dr. Dinush Jayasooriya for verifying calculations. I wish to thank the reviewers for their very helpful suggestions. I wish to especially thank Sabrina Lea Wille for constant encouragement to complete this manuscript after the author experienced a brain injury.

Author Contributions

Porter Eldridge Coggins is the sole author. The author read and approved the final manuscript.

Funding

This work was not supported by any external funding.

Data Availability Statement

Not applicable, no data was used for this research project.

Conflicts of Interest

The author declares no conflicts of interest.

References

[1] Hill, L. S. Cryptography in an Algebraic Alphabet. The American Mathematical Monthly. 1929. 36(6), 306-312. https://doi.org/10.2307.2298294

[2] Hill, L. S. Concerning Certain Linear Transformation Apparatus of Cryptography. The American Mathematical Monthly. 1931, 38(3), 135-154. https:/doi.org/10.1080/00029890.1931.11987161

[3] Sharma, N., Chirgaiya, S. A Review of Modern Hill Cipher Techniques. International Journal for Scientific Research & Development. 2013, 1(10), 2198-2202.

[4] Agrawal K., Gera A. Elliptic Curve Cryptography with Hill Cipher Generation for Secure Text Cryptosystem. International journal of computer applications. 2014 Jan 1, 106(1).

[5] Dawahdeh Z. E., Yaakob S. N., bin Othman R. R. A New Image Encryption Technique Combining Elliptic Curve Cryptosystem with Hill Cipher. Journal of King Saud University-Computer and Information Sciences. 2018 Jul 1, 30(3), 349-55. https://doi.org/10.1016/j.jksuci.2017.06.004

[6] Santoso YS. Message Security Using a Combination of Hill Cipher and RSA Algorithms. Jurnal Matematika Dan Ilmu Pengetahuan Alam LLDikti Wilayah 1 (JUMPA). 2021 Mar 30, 1(1), 20-8. https://doi.org/10.54076/jumpa.v1i1.38

[7] Arifin S., Kurniadi F. I., Yudistira. I. G., Nariswari R., Murnaka N. P., Muktyas I. B. Image Encryption Algorithm Through Hill Cipher, Shift 128 Cipher, and Logistic Map Using Python. In 2022 3rd International Conference on Artificial Intelligence and Data Sciences (AiDAS) 2022 Sep 7, 221-226. https://doi.org/10.1109/AiDAS56890.2022.9918696

[8] Wen H, Lin Y, Yang L, Chen R. Cryptanalysis of an Image Encryption Scheme using Variant Hill cipher and Chaos. Expert Systems with Applications. 2024, Sep 15, 250, 123748. https://doi.org/10.1016/j.eswa.2024.123748

[9] Hasoun R. K., Khlebus S. F., Tayyeh H. K. A New Approach of Classical Hill Cipher in Public Key Cryptography. International Journal of Nonlinear Analysis and Applications. 2021 Nov 1, 12(2), 1071-82. https://doi.org/10.22075/IJNAA.2021.5176

[10] Levine J, Nahikian HM. On the Construction of Involutory Matrices. The American Mathematical Monthly. 1962 Apr 1, 69(4), 267-72. https://doi.org/10.1080/00029890.1962.11989880

[11] Achary, B., Jena D., Patra S. K, Panda G. Invertilbe, Involutory and Permutation Matrix Generation Methods for Hill Cipher System. International Conference on Advanced Computer Control, Singapore, Singapore, 2009; 410-414. https://doi.org/10.1109/ICACC.2009.3

[12] Putera A, Siahaan U, Rahim R. Dynamic Key Matrix of Hill Cipher Using Genetic Algorithm. Int. J. Secur. Its Appl. 2016 Aug 1, 10(8), 173-80. https://doi.org/10.11591/ijaas.v6.i4.pp313-318

[13] Reddy KA, Vishnuvardhan B, Krishna AV. A Modified Hill Cipher Based on Circulant Matrices. Procedia Technology. 2012 Jan 1, 4, 114-8. https://doi.org/10.1016/j.protcy.2012.05.016

[14] Coggins III P. E., Glatzer T. An Algorithm for a Matrix-Based Enigma Encoder from a Variation of the Hill Cipher as an Application of 2×2 Matrices. Primus. 2020 Jan 2, 30(1), 1-8. https://doi.org/10.1080/10511970.2018.1493010

[15] Santoso H, Rambe NS, Suhardi S. Combined Performance of Hill Cipher and Rivest Code 6 (Rc6) Algorithms in Image Security. IJISTECH (International Journal of Information System and Technology). 2024 Apr 30, 7(6), 379-85. Akreditasi No. 158/E/KPT/2021.

[16] Hassan A., Garko A., Sani S., Abdullahi U., Sahalu S. Combined Techniques of Hill Cipher and Transposition Cipher. Journal of Mathematical Letters. 2023, 1(1), 57-64. https://doi.org/10.31586/jml.2023.822

[17] ElHabshy, A. A. Augmented Hill Cipher. International Journal of Network Security. Sept 2019, 21(5), 812-818. https:/doi.org/10.6633/IJNS.201909_21(5).13

[18] Rekha G., Srinivas V. A Novel Approach in Hill Cipher Cryptography. 2023, 11(6), 3503-3505. https://doi.org/10.47191/ijmcr/vaai6.06

[19] Khalaf AA, Abd El-karim MS, Hamed HF. A Triple Hill Cipher Algorithm Proposed to Increase the Security of Encrypted Binary Data and its Implementation Using FPGA. In2016 18th International Conference on Advanced Communication Technology (ICACT) 2016 Jan 31, 752-759.

[20] Sastry, V. U. K., Samson Ch. A Generalized Hill Cipher Involving Different Powers of a Key, Mixing and Substitution. International Journal of Advanced Research in Computer Science. July-August 2012, 3(4), 191-197. doi unavailable https://ijarcs.info/index.php/Ijarcs/article/view/1266/1254

[21] Shannon, C. E. Communication Theory of Secrecy Systems. Bell Systems Technical Journal. 1949, 28(4), 656-715.

[22] Hodges, J. H. The Matrix Equation X2 – I = 0 Over a Finite Field. The American Mathematical Monthly. Aug. – Sep., 1958, 65(7), 518-520. https://doi.org/10.2307/2308579

[23] Overbey, J., Traves, W., & Wojdylo, J. (2005). On The Keyspace Of The Hill Cipher. Cryptologia, 29(1), 59–72. https://doi.org/10.1080/0161-110591893771

Biography



Porter Eldridge Coggins holds the rank of Professor of Professional Education at Bemidji State University, Department of Professional Education where he formerly served as chair. He completed his three graduate degrees, PhD in Education (Human and Machine Cognition), Master of Science in Interdisciplinary Studies (Computational Linguistics), and Master of Arts in Teaching Mathematics all at from the University of Idaho. He has published various academic scholarly peer-reviewed, juried, refereed articles in mathematics, neuroscience, and cryptography. He holds professional memberships in ACM, IEEE, ACL, CCSC. He has served as a peer reviewer for many academic scholarly publications. He has served as Chair of the Department of Professional Education at Bemidji State University.

Research Field

Porter Eldridge Coggins, III: cryptography (Hill Cipher), genetic programming, machine cognition, game theory, neuroscience.