

Algebraic Approaches to Information Security: Applying Ring Matrix Groups for Enhanced Protection

Sisilia Sylviani^{1,*}, Nita Anastasya¹, Ema Carnia¹, Fahmi Candra Permana², and Nursanti Anggriani¹

¹Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Padjadjaran, Sumedang 45363, Indonesia

²Department of Software Engineering, UPI Campus in Cibiru, 40625, Universitas Pendidikan Indonesia, Indonesia

Received: 12 Nov. 2024, Revised: 22 Feb. 2025, Accepted: 15 Mar. 2025

Published online: 1 May 2025

Abstract: The group matrix ring $M_2(R)G$ serves as a generator matrix for constructing binary linear codes by integrating algebraic properties of groups and matrices. This research explores the application of $M_2(R)G$, where $M_2(R)$ denotes the set of 2×2 matrices with entries from a ring R , and G represents a specific group. The study investigates how this structure can be effectively utilized in generating binary linear codes through a generator matrix, highlighting its potential advantages in error correction and secure communication. The findings provide deeper insights into the theoretical framework of group matrix rings and their practical applications in coding theory, contributing to advancements in information security.

Keywords: Group matrix ring, binary linear code, generator matrix, information security

1 Introduction

Currently, information security has developed into something very important. As society increasingly relies on digital networks for communication, business and data storage, the need to prevent misuse of sensitive data or illegal access is increasing for both individuals and companies. Furthermore, with cyber threats continuing to grow in complexity and severity ([1], [2], [3], [4], [5]), strong information security is essential to protect against potential risks and vulnerabilities ([6], [7], [8]).

The amount of highly sensitive information stored digitally is rapidly growing; accordingly, businesses should implement tighter security to prevent unauthorized access. These provisions are necessary to prevent unauthorized access and misinformation. Numerous studies emphasize the importance of implementing strong security measures to protect information from breaches and errors. Alshboul [9] emphasizes the need for security measures, preventive measures, and policies that tailored to protect organizational assets from internal and external attacks. Kidd [10] highlights the need for regular security audits, intrusion detection, and a focus on human factors in information security.

Several studies have investigated methods for enhancing information security. Popa [11] and Arutyunov [12] both emphasize the potential of combining biometrics and cryptography to improve system security. Schaefer [13] highlights the role of information theory approaches in designing secure information systems. Additionally, algebraic techniques have also been explored, such as the use of group theory, matrix rings, and error-correcting codes to enhance the structural robustness of security mechanisms [14], [15], [16], [17], [18].

From the results obtained above, it follows that coding theory is one of such scientific branches that can aid in this process. Coding theory is a branch of mathematics as well as computer science that is concerned with the design of error-correcting codes. These algorithms are designed to find and fix errors which might occur due to noise, interference or other kinds of errors leading to data corruption. They are also used to protect from these kind of mistakes. A code is a representation of a collection of messages created in order to shield messages from mistakes during transportation. Several types of code include: linear codes, convolution code, and turbo code. A linear code of length n is a code where its alphabet is comprised of a finite field. One example is binary linear

* Corresponding author e-mail: sisilia.sylviani@unpad.ac.id

code, which is a code with elements ranging from fields to 2 members, namely 0 and 1. Linear code is a basic concept in coding theory, with various applications in error correction and data transmission. Muller et al [19] introduced the concept of coding computing, which aims to reduce the computational cost of matrix-vector multiplication. Additionally, [20] discusses linear codes derived from projective spaces.

Binary codes are extensively used in the telecommunications, digital communication systems, as well as computer networks in the data transmission. They allow reliable and efficient to be occur at any errors that may appear during the transmission of information through the channels, which are noise. Binary codes are most commonly used in digital storage systems, such as: Hard drives. This assures the integrity and accuracy of stored data, for example, in the existence of storage media errors. Binary codes are models in which cyphers have important varieties in cryptography and information defense. Error correction codes are applied to prevent corruption and interference of data especially sensible data during data transmission and data storage.

Binary linear code can be built or constructed by various methods ([21], [22]). Some of them are Concatenation, Parity Check Matrix, Reed-Solomon Codes, Convolutional Codes, and Generator Matrix. This article will focus on discussing the formation of binary linear codes using a generator matrix from the group matrix ring $M_k(R)G$. The group matrix ring $M_k(R)G$ is a set of matrices with entries in the matrix as elements of the group ring RG , where k represents the size of the matrix forming a $k \times k$ block. The ring group itself, denoted by RG , constitutes a formal series or quantity formed from each ring element and its finite group of elements. A matrix from this ring can serve as a generator for constructing binary linear codes. This paper mainly focuses on studying the role of the group matrix ring $M_2(R)G$ in constructing binary linear codes.

2 Methodologies

The initial stage in this study is to choose a finite group G . In this case, dihedral group D_8 is selected to be the designated group. It is a finite group with 16 elements. Dihedral group D_8 , also known as the octagonal symmetry group, is a symmetry group consisting of all symmetry transformations that preserve the octagon shape but can change its position and orientation. This group is a well-known examples of the important symmetry group, is often utilised in geometric, crystallographic and computer graphic areas for modeling symmetry and transformations in two dimensional shapes, like octagons and other shapes. The two views of a standard octagon appear in Figure 1. The figure contains two parts which show the shape in figure (a) and the numbered vertices in figure (b). The labeling system

identifies symmetrical patterns in the octagon that demonstrate the dihedral group D_8 structure.

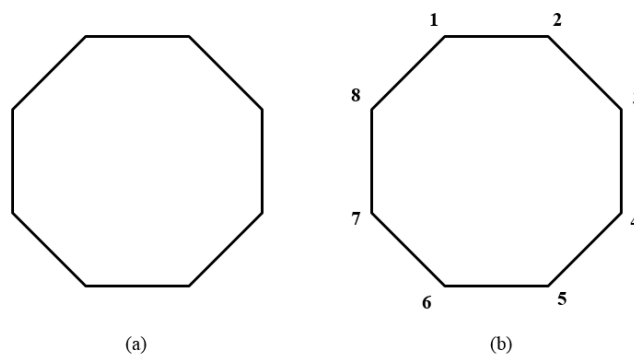


Fig. 1: (a) Octagon (b) Octagon with numbering

The elements in the dihedral group D_8 consist of rotations and reflections. Rotation refers to the act of rotating clockwise or counterclockwise around the center of the octagon at angles such as 45 degrees, 90 degrees, 135 degrees, and so forth. Following the group selection, the next step is to determine the mapping $\tau_k(v)$, which sends an element v from the matrix ring $M_k(R)G$ to $k \times k$ matrices over the ring R . This mapping is defined in the context of the chosen group G and the form of $k \times k$ matrices over R . The group matrix ring denoted by $M_k(R)G$ is a set of matrices whose entries in the matrix are elements of the group ring RG .

Let G be any finite group and let R be any ring with unity. The group ring RG set is formed with its members being the formal sum $\sum_{i \in I} r_i g_i$ where $r_i \in R$ and $g_i \in G$ [23]. The group matrix ring $M_k(R)G$ is a $k \times k$ matrix with entries from ring R [24].

$$M_k(R)G = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1k} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{k1} & a_{k2} & a_{k3} & \cdots & a_{kk} \end{pmatrix}.$$

The index k in $M_k(R)G$ is the block matrix size of the group matrix ring. The overall matrix size could be much larger. Let $v = \alpha_{g_1} g_1 + \alpha_{g_2} g_2 + \alpha_{g_3} g_3 + \dots + \alpha_{g_n} g_n \in RG$. Let $\sigma(v) \in M_k(R)G$ be matrix described as follows

$$\sigma(v) = \begin{pmatrix} \alpha_{g_1^{-1}g_1} & \alpha_{g_1^{-1}g_2} & \alpha_{g_1^{-1}g_3} & \cdots & \alpha_{g_1^{-1}g_n} \\ \alpha_{g_2^{-1}g_1} & \alpha_{g_2^{-1}g_2} & \alpha_{g_2^{-1}g_3} & \cdots & \alpha_{g_2^{-1}g_n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_{g_n^{-1}g_1} & \alpha_{g_n^{-1}g_2} & \alpha_{g_n^{-1}g_3} & \cdots & \alpha_{g_n^{-1}g_n} \end{pmatrix}$$

where $g_1^{-1}, g_2^{-1}, g_3^{-1}, \dots, g_n^{-1} \in G$. the mapping σ is defined as follow.

$$\sigma : RG \rightarrow M_k(R)G,$$

where the members of RG are mapped to matrices in $M_k(R)G$.

In this article, the value of k is set to 2. For this reason, the group ring matrix is $M_2(R)D_8$ with the mapping represented by $\tau_2(v)$, which maps an element from $M_2(R)D_8$ to a 2×2 matrix over the ring R . Subsequently, the research involves determining generator matrices in the form of $[I_{2n} | \tau_2(v)]$, where I_{kn} represents the identity matrix of size $2n \times 2n$, and v is an element in the matrix ring $M_2(R)G$. The next phase employs these generator matrices to directly find self-dual binary codes over the finite field F_2 .

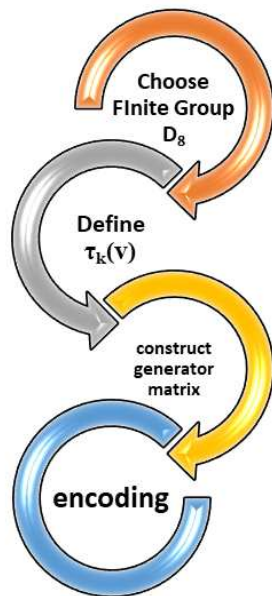


Fig. 2: The research steps

By applying the above methodology—from selecting the dihedral group D_8 and mapping its elements into the matrix ring $M_2(R)$, to constructing and testing generator matrices—this research aims to leverage the algebraic robustness of group and ring theory for enhanced code design and can be optimized for data security. The subsequent section, Results and Discussion, will detail the empirical outcomes of this method, highlighting both the theoretical significance and practical implications of employing dihedral symmetries in binary linear coding.

3 Results and Discussions

This section presents a detailed examination of the codes generated using the dihedral group D_8 . The discussion

begins by illustrating how the group’s rotational and reflective elements translate into distinct matrix representations. Throughout this analysis, the manner in which dihedral symmetries influence code structure is emphasized, ultimately illustrating the potential advantages of leveraging group-theoretic properties for robust and efficient error correction in information security contexts.

The first step taken is to obtain the matrix representation of the elements of D_8 . As mentioned in the previous section, the dihedral group D_8 is known as the dihedral group of order 16, encompassing the symmetries of a regular octagon. Specifically, the group contains eight rotation elements and eight reflection elements, each of which can be expressed in terms of a 2×2 transformation matrix that acts on the plane containing the octagon. The rotation elements correspond to angular rotations by multiples of 45° . These can be written as:

$$R_k = \begin{pmatrix} \cos \frac{k\pi}{4} & -\sin \frac{k\pi}{4} \\ \sin \frac{k\pi}{4} & \cos \frac{k\pi}{4} \end{pmatrix}, \quad k = 0, 1, \dots, 7,$$

and are listed in Table 1 with their corresponding degrees of rotation. Each rotation effectively repositions the vertices of the octagon around its center, preserving distances and angles, thus highlighting the key geometric symmetry that defines D_8 . Furthermore, the reflection element is defined as a reflection about the axis of symmetry that passes through the midpoint of each side of the octagon or a reflection about the axis of symmetry that passes through two corner points. For more details, see Figure 3.

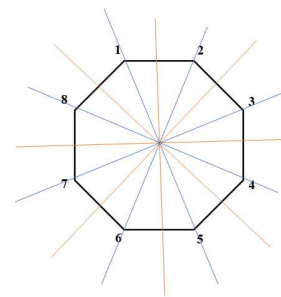


Fig. 3: Reflection of each side and corner point of the octagon

The blue line is a reflection of each corner and the orange line is a reflection of each side. First, it is reflected on each side as follows. The reflection elements of D_8 are shown in detail in Table 2.

Table 1: All rotation elements of D_8

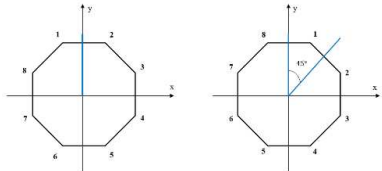
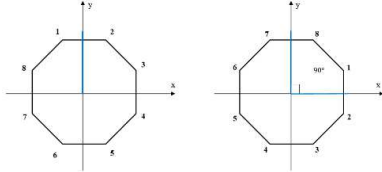
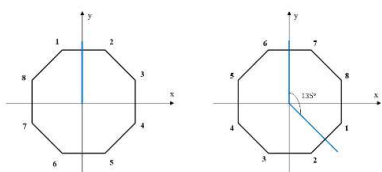
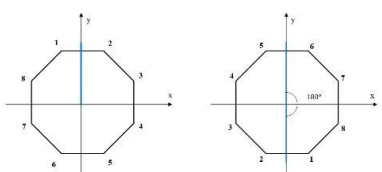
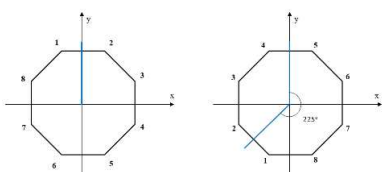
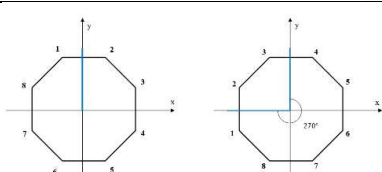
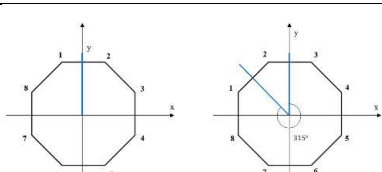
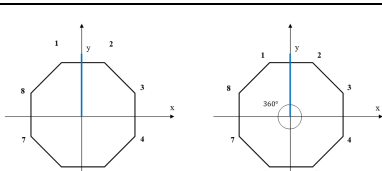
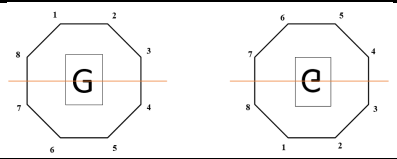
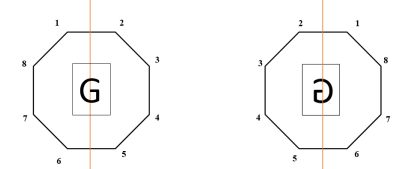
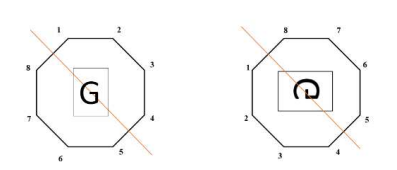
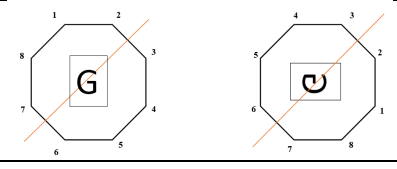
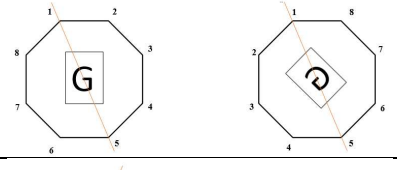
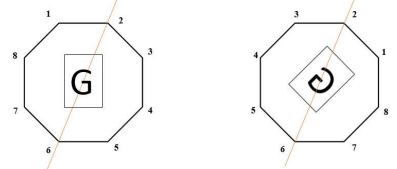
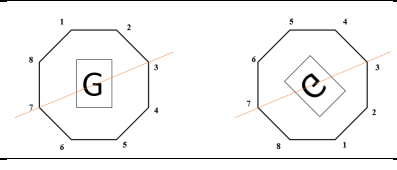
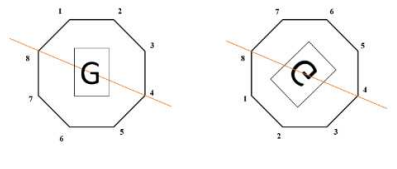
No	Rotation elements	Figure	Permutation
1	45		$r_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 1 \end{pmatrix} = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$
2	90		$r_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 4 & 5 & 6 & 7 & 8 & 1 & 2 \end{pmatrix} = (1\ 3\ 5\ 7)(2\ 4\ 6\ 8)$
3	135		$r_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 4 & 5 & 6 & 7 & 8 & 1 & 2 & 3 \end{pmatrix} = (1\ 4\ 7\ 2\ 5\ 8\ 3\ 6)$
4	180		$r_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 5 & 6 & 7 & 8 & 1 & 2 & 3 & 4 \end{pmatrix} = (1\ 5)(2\ 6)(3\ 7)(4\ 8)$
5	225		$r_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 6 & 7 & 8 & 1 & 2 & 3 & 4 & 5 \end{pmatrix} = (1\ 6\ 3\ 8\ 5\ 2\ 7\ 4)$
6	270		$r_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 7 & 8 & 1 & 2 & 3 & 4 & 5 & 6 \end{pmatrix} = (1\ 7\ 5\ 3)(2\ 8\ 6\ 4)$
7	315		$r_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 8 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix} = (1\ 8\ 7\ 6\ 5\ 4\ 3\ 2)$
8	360		$r_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{pmatrix} \\ = (1)(2)(3)(4)(5)(6)(7)(8) = (1)$

Table 2: All reflection elements of D_8

No	Reflection elements	Figure	Permutation
1	Horizontal reflection		$h = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 6 & 5 & 4 & 3 & 2 & 1 & 8 & 7 \end{pmatrix} = (1\ 6)(2\ 5)(3\ 4)(7\ 8)$
2	Vertical reflection		$v = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 1 & 8 & 7 & 6 & 5 & 4 & 3 \end{pmatrix} = (1\ 2)(3\ 8)(4\ 7)(5\ 6)$
3	Diagonal reflection		$d = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \end{pmatrix} = (1\ 8)(2\ 7)(3\ 6)(4\ 5)$
4	Invers diagonal reflection		$d' = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 4 & 3 & 2 & 1 & 8 & 7 & 6 & 5 \end{pmatrix} = (1\ 4)(2\ 3)(5\ 8)(6\ 7)$
5	Reflection of corner points 1 and 5		$a_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 8 & 7 & 6 & 5 & 4 & 3 & 2 \end{pmatrix} = (2\ 8)(3\ 7)(4\ 6)$
6	Reflection of corner points 2 and 6		$a_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 2 & 1 & 8 & 7 & 6 & 5 & 4 \end{pmatrix} = (1\ 3)(4\ 8)(5\ 7)$
7	Reflection of corner points 3 and 7		$a_3 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 5 & 4 & 3 & 2 & 1 & 8 & 7 & 6 \end{pmatrix} = (1\ 5)(2\ 4)(6\ 8)$
8	Reflection of corner points 4 and 8		$a_4 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 7 & 6 & 5 & 4 & 3 & 2 & 1 & 8 \end{pmatrix} = (1\ 7)(2\ 6)(3\ 5)$

Based on the above explanation, it can be seen that the dihedral group D_8 consists of 16 elements, representing all symmetries of a regular octagon, or it can be written as follow

$$D_8 = \left\{ r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8 \right\} \cup \left\{ h, v, d, d', \alpha_1, \alpha_2, \alpha_3, \alpha_4 \right\}.$$

The operation in the dihedral group D_8 is a transformation composition, in which two transformations are sequentially combined to produce a new transformation. For example, if r_1 is a 45 degree rotation and h is a horizontal reflection, the composition of $r_1 h$ resulting in a reflection angle of 4.8 is α_4 . For more details, below is an illustration of the composition of $r_1 h$.

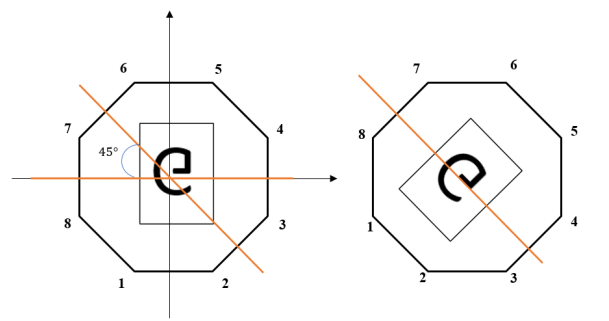


Fig. 5: Illustration of 45 degree rotation resulting from horizontal reflection

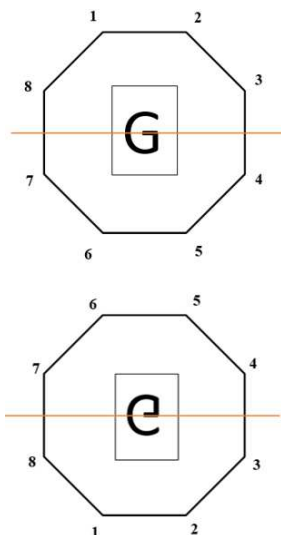


Fig. 4: Horizontal reflection

The composition permutation operation begins with a horizontal reflection of the octagon, as shown in Figure 4. Following this reflection, the resulting transformation is rotated by 45 degrees, as illustrated in Figure 5. In Figure 4, point 1 is mapped to point 6. Subsequently, in Figure 5, point 1 moves again, this time to point 8, where point 8 coincides with point 7 in the final position. Thus, in the permutation operation involving the composition of r_1 and h , point 1 is ultimately mapped to point 7. This process is applied consistently to every point in the octagon, resulting in the permutation α_4 , which corresponds to a reflection of corner points 4 and 8. Consequently, the composition of the permutation r_1 and h yields the following results.

$$\begin{aligned} r_1 \circ h &= (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8) \circ (1\ 6)\ (2\ 5)\ (3\ 4)\ (7\ 8) \\ &= (1\ 7)\ (2\ 6)\ (3\ 5) \\ &= \alpha_4. \end{aligned}$$

The dihedral group D_8 can also be depicted using the Cayley diagram and group table as shown by Table 3.

Table 3: Cayley Table for (D_8, \circ)

\circ	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	h	v	d	d'	α_1	α_2	α_3	α_4
r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_1	α_4	α_2	α_1	α_3	v	d'	h	d
r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_1	r_2	d	d'	v	h	α_2	α_3	α_4	α_1
r_3	r_4	r_5	r_6	r_7	r_8	r_1	r_2	r_3	α_1	α_3	α_2	α_4	d'	h	d	v
r_4	r_5	r_6	r_7	r_8	r_1	r_2	r_3	r_4	v	h	d'	d	α_3	α_4	α_1	α_2
r_5	r_6	r_7	r_8	r_1	r_2	r_3	r_4	r_5	α_2	α_4	α_3	α_1	h	d	v	d'
r_6	r_7	r_8	r_1	r_2	r_3	r_4	r_5	r_6	d'	d	h	v	α_4	α_1	α_2	α_3
r_7	r_8	r_1	r_2	r_3	r_4	r_5	r_6	r_7	α_3	α_1	α_4	α_2	d	v	d'	h
r_8	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	h	v	d	d'	α_1	α_2	α_3	α_4
h	α_3	d'	α_2	v	α_1	d	α_4	h	r_8	r_4	r_6	r_2	r_5	r_3	r_1	r_7
v	α_1	d	α_4	h	α_3	d'	α_2	v	r_4	r_8	r_2	r_6	r_1	r_7	r_5	r_3
d	α_4	h	α_3	d'	α_2	v	α_1	d	r_2	r_6	r_8	r_4	r_7	r_5	r_3	r_1
d'	α_2	v	α_1	d	α_4	h	α_3	d'	r_6	r_2	r_4	r_8	r_3	r_1	r_7	r_5
α_1	d	α_4	h	α_3	d'	α_2	v	α_1	r_3	r_7	r_1	r_5	r_8	r_6	r_4	r_2
α_2	v	α_1	d	α_4	h	α_3	d'	α_2	r_5	r_1	r_3	r_7	r_2	r_8	r_6	r_4
α_3	d'	α_2	v	α_1	d	α_4	h	α_3	r_7	r_3	r_5	r_1	r_4	r_2	r_8	r_6
α_4	h	α_3	d'	α_2	v	α_1	d	α_4	r_1	r_5	r_7	r_3	r_6	r_4	r_2	r_8

The elements of D_n can be represented as a 2×2 matrix, with the group operation corresponding to the matrix multiplication operation. In particular,

$$R_k = \begin{pmatrix} \cos(2\pi k/n) & -\sin(2\pi k/n) \\ \sin(2\pi k/n) & \cos(2\pi k/n) \end{pmatrix},$$

$$S_k = \begin{pmatrix} \cos(2\pi k/n) & \sin(2\pi k/n) \\ \sin(2\pi k/n) & -\cos(2\pi k/n) \end{pmatrix}.$$

The Matrix R_k defined as a rotation matrix representing clockwise rotation through an angle of $2\pi k/n$. S_k is a reflection matrix that crosses a line that forms an angle $\pi k/n$ with the x axis. Below are several examples of representation of D_8 elements in a 2×2 matrix.

Table 4: Matrix Representation of Elements of D_8

Elements of D_8	Matrix Representation
90 degree rotation	$\begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$
180 degree rotation	$\begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$
270 degree rotation	$\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$
360 degree rotation	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
Horizontal reflection. The line of intersection has 0 degrees magnitude with the x -axis.	$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$
Vertical reflection. The line of intersection is 90 degrees to the x -axis.	$\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$
Diagonal reflection. The intersection line has a magnitude of 45 degrees with the x -axis.	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
Inverse diagonal reflection. The line of intersection has a magnitude of 135 degrees with the x -axis	$\begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}$

Consider $M_2(R)G$, a group of order 18 and define the transformation $\tau_2(v)$ as a 2×2 block matrix [24]:

$$\tau_2(v_1) = \begin{pmatrix} A & B \\ B & A \end{pmatrix}.$$

The matrix A and the matrix B are defined as follows.

$$A = CIRC(A_1, A_2, A_3, \dots, A_9),$$

$$B = CIRC(A_{10}, A_{11}, A_{12}, \dots, A_{18}),$$

where $A_i \in M_2(R)$.

$$\tau_2(v) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Example 4.4 [23] Let

$$v = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} a + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} a^2 + \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} a^3 + \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} b + \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} ba + \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} ba^2 + \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} ba^2 \in M_2(\mathbb{F})D_8,$$

where $\langle a, b \rangle \cong D_8$, dihedral group with 8 elements. Then $\sigma_2(v)$ generates the code $C_2(v)$.

Next, the properties of the matrix $\tau_2(a)$ are examined. The matrix $\tau_2(a)$ consists of the following elements.

$$v_1 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, v_2 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$

$$v_3 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, v_4 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

$$v_5 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, v_6 = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix},$$

$$v_7 = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, v_8 = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}.$$

Thus the matrix $\tau_2(v)$ can be expressed in the following form

$$\tau_2(v) = \begin{pmatrix} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 \\ v_4 & v_1 & v_2 & v_3 & v_8 & v_5 & v_6 & v_7 \\ v_3 & v_4 & v_1 & v_2 & v_7 & v_8 & v_5 & v_6 \\ v_2 & v_3 & v_4 & v_1 & v_6 & v_7 & v_8 & v_5 \\ v_5 & v_6 & v_7 & v_8 & v_1 & v_4 & v_2 & v_3 \\ v_8 & v_5 & v_6 & v_7 & v_3 & v_1 & v_4 & v_2 \\ v_7 & v_8 & v_5 & v_6 & v_2 & v_3 & v_1 & v_4 \\ v_6 & v_7 & v_8 & v_5 & v_4 & v_2 & v_3 & v_1 \end{pmatrix}.$$

Then the matrix $\tau_2(v)$ above, it can be seen that this matrix is a block matrix. Here are the partitions.

$$\tau_2(v) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$= \begin{pmatrix} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 \\ v_4 & v_1 & v_2 & v_3 & v_8 & v_5 & v_6 & v_7 \\ v_3 & v_4 & v_1 & v_2 & v_7 & v_8 & v_5 & v_6 \\ v_2 & v_3 & v_4 & v_1 & v_6 & v_7 & v_8 & v_5 \\ v_5 & v_6 & v_7 & v_8 & v_1 & v_4 & v_2 & v_3 \\ v_8 & v_5 & v_6 & v_7 & v_3 & v_1 & v_4 & v_2 \\ v_7 & v_8 & v_5 & v_6 & v_2 & v_3 & v_1 & v_4 \\ v_6 & v_7 & v_8 & v_5 & v_4 & v_2 & v_3 & v_1 \end{pmatrix} = \begin{pmatrix} A & B \\ B & C \end{pmatrix}.$$

$$\tau_2(v) = CIRC(A, B, C)$$

The submatrix is defined follows.

1. Submatrix A

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$= \begin{pmatrix} v_1 & v_2 & v_3 & v_4 \\ v_4 & v_1 & v_2 & v_3 \\ v_3 & v_4 & v_1 & v_2 \\ v_2 & v_3 & v_4 & v_1 \end{pmatrix}.$$

Based on the description above, it can be seen that submatrix A is a 4×4 circular matrix or can be denoted as

$$A = circ(v_1, v_2, v_3, v_4).$$

Next, transpose matrix A as follows.

$$A^T = \begin{pmatrix} v_1 & v_4 & v_3 & v_2 \\ v_2 & v_1 & v_4 & v_3 \\ v_3 & v_2 & v_1 & v_4 \\ v_4 & v_3 & v_2 & v_1 \end{pmatrix}$$

It can be seen that the transpose of submatrix A is not the same as matrix A itself, thus submatrix A is not a

symmetric matrix. Then submatrix A is a 4×4 persymmetric matrix. For further details, the form of the subtranspose matrix is first determined. The following is the calculation of each submatrix entry of A into a subtranspose matrix entry of submatrix A.

$$a_{ij} \rightarrow a_{n-j+1, n-i+1}^S.$$

Table 5: Mapping of Submatrix A Entries to Submatrix A^S Entries

Submatrix A entries	Submatrix entry A ^S
a_{11}	$a_{4-1+1, 4-1+1}^S = a_{44}^S$
a_{12}	$a_{4-2+1, 4-1+1}^S = a_{34}^S$
a_{13}	$a_{4-2+1, 4-1+1}^S = a_{24}^S$
a_{14}	$a_{4-4+1, 4-1+1}^S = a_{14}^S$
a_{21}	$a_{4-1+1, 4-2+1}^S = a_{43}^S$
a_{22}	$a_{4-2+1, 4-2+1}^S = a_{33}^S$
a_{23}	$a_{4-2+1, 4-2+1}^S = a_{23}^S$
a_{24}	$a_{4-4+1, 4-2+1}^S = a_{13}^S$
a_{31}	$a_{4-1+1, 4-3+1}^S = a_{42}^S$
a_{32}	$a_{4-2+1, 4-3+1}^S = a_{32}^S$
a_{33}	$a_{4-2+1, 4-3+1}^S = a_{22}^S$
a_{34}	$a_{4-4+1, 4-3+1}^S = a_{12}^S$
a_{41}	$a_{4-1+1, 4-4+1}^S = a_{41}^S$
a_{42}	$a_{4-2+1, 4-4+1}^S = a_{31}^S$
a_{43}	$a_{4-2+1, 4-4+1}^S = a_{21}^S$
a_{44}	$a_{4-4+1, 4-4+1}^S = a_{11}^S$

Based on the calculation above, the subtranspose matrix is

$$A^S = \begin{pmatrix} v_1 & v_2 & v_3 & v_4 \\ v_4 & v_1 & v_2 & v_3 \\ v_3 & v_4 & v_1 & v_2 \\ v_2 & v_3 & v_4 & v_1 \end{pmatrix}.$$

Thus, it can be seen that the subtranspose matrix is the same as submatrix A. Hence, submatrix A is a persymmetric matrix, it can be denoted as

$$A = persym(v_1, v_2, v_3, v_4).$$

2. Submatrix B

$$b = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{pmatrix} = \begin{pmatrix} v_5 & v_6 & v_7 & v_8 \\ v_8 & v_5 & v_6 & v_7 \\ v_7 & v_8 & v_5 & v_6 \\ v_6 & v_7 & v_8 & v_5 \end{pmatrix}.$$

Based on the description above, it can be seen that submatrix B is a 4×4 circular matrix or can be denoted as

$$B = circ(v_5, v_6, v_7, v_8).$$

Next, transpose matrix B as follows.

$$B^T = \begin{pmatrix} v_5 & v_8 & v_7 & v_6 \\ v_6 & v_5 & v_8 & v_7 \\ v_7 & v_6 & v_5 & v_8 \\ v_8 & v_7 & v_6 & v_5 \end{pmatrix}.$$

It can be seen that the transpose of submatrix B is not the same as matrix B itself, thus submatrix B is not a symmetric matrix. Then, submatrix B is a 4×4 persymmetric matrix. For further details, the form of the subtranspose matrix is first determined. The following is the calculation of each submatrix entry from B to become a subtranspose matrix entry from submatrix B .

$$b_{ij} \rightarrow b_{n-j+1, n-i+1}^S.$$

Table 6: Mapping of submatrix B entries become the B^S submatrix entries

Submatrix B entries	Submatrix entry B^S
b_{11}	$b_{4-1+1, 4-1+1}^S = b_{44}^S$
b_{12}	$b_{4-2+1, 4-1+1}^S = b_{34}^S$
b_{13}	$b_{4-2+1, 4-1+1}^S = b_{24}^S$
b_{14}	$b_{4-4+1, 4-1+1}^S = b_{14}^S$
b_{21}	$b_{4-1+1, 4-2+1}^S = b_{43}^S$
b_{22}	$b_{4-2+1, 4-2+1}^S = b_{33}^S$
b_{23}	$b_{4-2+1, 4-2+1}^S = b_{23}^S$
b_{24}	$b_{4-4+1, 4-2+1}^S = b_{13}^S$
b_{31}	$b_{4-1+1, 4-3+1}^S = b_{42}^S$
b_{32}	$b_{4-2+1, 4-3+1}^S = b_{32}^S$
b_{33}	$b_{4-2+1, 4-3+1}^S = b_{22}^S$
b_{34}	$b_{4-4+1, 4-3+1}^S = b_{12}^S$
b_{41}	$b_{4-1+1, 4-4+1}^S = b_{41}^S$
b_{42}	$b_{4-2+1, 4-4+1}^S = b_{31}^S$
b_{43}	$b_{4-2+1, 4-4+1}^S = b_{21}^S$
b_{44}	$b_{4-4+1, 4-4+1}^S = b_{11}^S$

Based on the calculation above, the subtranspose matrix is

$$B^S = \begin{pmatrix} v_5 & v_6 & v_7 & v_8 \\ v_8 & v_5 & v_6 & v_7 \\ v_7 & v_8 & v_5 & v_6 \\ v_6 & v_7 & v_8 & v_5 \end{pmatrix}.$$

Thus, it can be seen that the subtranspose matrix is the same as submatrix B . Hence, submatrix B is a persymmetric matrix, it can be denoted as

$$B = \text{persym}(v_5, v_6, v_7, v_8).$$

3. Submatrix C

$$C = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} v_1 & v_4 & v_2 & v_3 \\ v_3 & v_1 & v_4 & v_2 \\ v_2 & v_3 & v_1 & v_4 \\ v_4 & v_2 & v_3 & v_1 \end{pmatrix}.$$

Based on the description above, it can be seen that submatrix C is a 4×4 circular matrix or can be denoted as

$$C = \text{circ}(v_1, v_4, v_2, v_3).$$

Next, transpose the matrix C as follows.

$$C^T = \begin{pmatrix} v_1 & v_3 & v_2 & v_4 \\ v_4 & v_1 & v_3 & v_2 \\ v_2 & v_4 & v_1 & v_3 \\ v_3 & v_2 & v_4 & v_1 \end{pmatrix}$$

It can be seen that the transpose of the submatrix C is not the same as the matrix C itself, thus the submatrix C is not a symmetric matrix. The submatrix C is also a persymmetric matrix 4×4 . For further details, the form of the subtranspose matrix is first determined. The following is the calculation of each submatrix entry of C into a subtranspose matrix entry of the submatrix C .

$$c_{ij} \rightarrow c_{n-j+1, n-i+1}^S.$$

Table 7: Mapping of submatrix C entries become the C^S submatrix entries

Submatrix C entries	Submatrix entry C^S
c_{11}	$c_{4-1+1, 4-1+1}^S = c_{44}^S$
c_{12}	$c_{4-2+1, 4-1+1}^S = c_{34}^S$
c_{13}	$c_{4-2+1, 4-1+1}^S = c_{24}^S$
c_{14}	$c_{4-4+1, 4-1+1}^S = c_{14}^S$
c_{21}	$c_{4-1+1, 4-2+1}^S = c_{43}^S$
c_{22}	$c_{4-2+1, 4-2+1}^S = c_{33}^S$
c_{23}	$c_{4-2+1, 4-2+1}^S = c_{23}^S$
c_{24}	$c_{4-4+1, 4-2+1}^S = c_{13}^S$
c_{31}	$c_{4-1+1, 4-3+1}^S = c_{42}^S$
c_{32}	$c_{4-2+1, 4-3+1}^S = c_{32}^S$
c_{33}	$c_{4-2+1, 4-3+1}^S = c_{22}^S$
c_{34}	$c_{4-4+1, 4-3+1}^S = c_{12}^S$
c_{41}	$c_{4-1+1, 4-4+1}^S = c_{41}^S$
c_{42}	$c_{4-2+1, 4-4+1}^S = c_{31}^S$
c_{43}	$c_{4-2+1, 4-4+1}^S = c_{21}^S$
c_{44}	$c_{4-4+1, 4-4+1}^S = c_{11}^S$

Based on the calculation above, the subtranspose matrix is

$$C^S = \begin{pmatrix} v_1 & v_4 & v_2 & v_3 \\ v_3 & v_1 & v_4 & v_2 \\ v_2 & v_3 & v_1 & v_4 \\ v_4 & v_2 & v_3 & v_1 \end{pmatrix}.$$

Hence, it follows that the subtranspose matrix is identical to the submatrix C . Consequently, C is a persymmetric matrix, which can be denoted as follows:

$$C = \text{persym}(v_1, v_4, v_2, v_3).$$

From the description above it can be seen that the matrix $\tau_2(v)$ is a block-circulant matrix, where the matrix $\tau_2(v)$ is a 2×2 block matrix, each submatrix is a circulant matrix and a persymmetric matrix.

Hence, from the above discussion, it can be observed that the matrix $\tau_2(v)$ exhibits a block-circulant structure, where each 2×2 block is both circulant and persymmetric. These properties provide a strong algebraic foundation for the subsequent construction and analysis of binary codes, motivating the next step in which the *generator matrix* is introduced and utilized to systematically encode information.

Generator Matrix

The generator matrix plays a central role in creating binary linear codes. For an (n, k) -code C , a generator matrix G is a $k \times n$ matrix whose rows form a basis for C . By multiplying G with a message vector (over \mathbb{F}_2 or another appropriate field), one obtains the encoded codeword. In this section, the way in which this construction leverages the structural properties identified in $\tau_2(v)$ to yield efficient and robust error-correcting codes is explored. The generator matrix is used to obtain the binary linear code. A generator matrix G for (n, k) -code C is a $k \times n$ matrix whose rows are the bases of C .

Example 1. Given $(5, 3)$ -code C with the following basis:

$$V_1 = (10001),$$

$$V_2 = (00010),$$

$$V_3 = (00100).$$

$$C = \{10001, 00010, 00100, 10010, 10100, 00110, 10111, 00000\}.$$

$$G = \begin{pmatrix} V_1 \\ V_2 \\ V_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

G is the generator matrix for $(5, 3)$ -C code. If the message $m = (x, y, z)$ is encoded to

$$mG = (x \ y \ z) \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} = (x \ 0 \ z \ y \ x).$$

The message to be received, namely $(x, 0, z, y, x)$ The $G(n, k)$ -code generator matrix C is said to be equal in standard form if G is of the form

$$G = (I_k \ A).$$

The matrix I_k is an identity matrix of size $k \times k$:

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}_{k \times k}.$$

Matrix A is a matrix of size $k \times (n - k)$.

Example 2. Looking at Example 1, if the chosen base is different, for example $V_1 = (10010)$,

$$V_2 = (01011),$$

$$V_3 = (00101),$$

Therefore, the matrix G takes the following form:

$$G = \begin{pmatrix} V_1 \\ V_2 \\ V_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix},$$

then the message $m = (x, y, z)$ is encoded as

$$\begin{aligned} mG &= (x \ y \ z) \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix} \\ &= (x \ y \ z \ x + y \ y + z). \end{aligned}$$

Therefore, the message to be received, viz $(x, y, z, x + y, y + z)$.

Example 3. [23] Let $v_1 = 1011$, $v_2 = 0101$ and generator matrix

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix},$$

The codeword $c_1 = v_1 G = 1011000$, $c_2 = v_2 G = 0101101 \in C$ is obtained.

Example 4. The generator matrix G is as follows

$$H = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}.$$

C is a $(4, 2)$ -code,

$$C = \{xH \in F_2^4 \mid x \in F_2^2\} = \{xH \in F_2^4 \mid x \in F_2^2\}.$$

All codewords obtained from the generator matrix H will be determined.

First, the initial code is determined, since the C code has $k = 2$, the initial code consists of 2 bits with members from F_2 . The initial code here is denoted by

$$\begin{aligned} \mathbf{x} &\in F_2^2. \\ \mathbf{x} &= \{00, 01, 10, 11\}. \end{aligned}$$

Then, to obtain the codewords, the initial code x is operated with the generator matrix H . The following is the explanation.

$$\begin{aligned} \mathbf{x}_1 = 00; \mathbf{c}_1 = \mathbf{x}_1 H &= (0 \ 0) \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \\ &= 0000; \\ \mathbf{x}_2 = 01; \mathbf{c}_2 = \mathbf{x}_2 H &= (0 \ 1) \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \\ &= 1100; \\ \mathbf{x}_3 = 10; \mathbf{c}_3 = \mathbf{x}_3 H &= (1 \ 0) \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \\ &= 0110; \\ \mathbf{x}_4 = 11; \mathbf{c}_4 = \mathbf{x}_4 H &= (1 \ 1) \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \\ &= 1010. \end{aligned}$$

Thus, 4 codewords are obtained in code C

$$C = \{0000, 1100, 0110, 1010\}.$$

Code Construction of Group Group Matrix Ring $M_2(R)G$

Building on the previous discussion about generator matrices, this subsection provides a detailed overview of how to construct a binary linear code using a generator matrix derived from the group matrix ring $M_2(R)G$. The key steps involved are as follows:

1. Vector Representation of Messages. A message is initially represented as a row vector of length k , where each element is a bit in F_2 . For example, let

$$v = \{v_1, v_2, v_3, \dots, v_k\} \quad \text{with } v \in F_2^k.$$

This vector v encapsulates the k -bit message that is to be encoded.

2. Vector Multiplication with a Generator Matrix. In order to produce the codeword c , the message vector v is multiplied by the generator matrix G :

$$c = vG.$$

Since operations take place in the finite field F_2 , all additions and multiplications are performed in modulo 2.

3. Computation Details. The multiplication vG is carried out by multiplying each element of v with the corresponding entry in each column of G , then summing (mod 2). Concretely:

- (a) Multiply each bit v_i in v by the corresponding entry in the current column of G .
- (b) Sum these products and reduce mod 2 to obtain one bit of the resulting codeword.
- (c) Move to the next column of G and repeat, ultimately forming the complete codeword c .

For instance, to multiply v by the first column of G , the following computation is performed:

$$\sum_{i=1}^k (v_i \times G_{i,1}) \pmod{2}.$$

Repeating this for each column produces the final codeword c .

A codeword c is the binary output obtained by multiplying a message vector v by a generator matrix G . In the context of binary linear codes, G is designed to ensure that each distinct k -bit message v maps to a unique and identifiable codeword c . This uniqueness is crucial for reliable decoding and error correction, as it guarantees that no two message words collide into the same codeword.

By applying the generator matrix, one transforms the original message into a structured binary sequence that inherently contains redundancy and error-detection/correction capabilities. Since each column (or row, depending on convention) of G encodes a portion of the message, the resulting codeword incorporates systematic patterns that aid in detecting and correcting transmission errors. Furthermore, during decoding, these patterns allow standard algorithms (e.g., syndrome-based decoders) to pinpoint and rectify corrupted bits, enabling successful retrieval of the original message v .

The following is a concise explanation intended to simplify the understanding of how codewords are generated. By outlining the fundamental steps, it aims to guide readers through the formation of binary codes from an original message. For larger code sizes, manual calculations can become exceedingly complex; hence, computational tools such as *Python* are recommended to streamline the process and ensure more efficient, organized computation.

1. Determining the initial code. Since the matrix $\tau_2(v)$ is 16×16 , each resulting codeword also has length 16, leading to $2^{16} = 65,536$ possible codewords. For demonstration purposes in this article, however, only a random subset of 20 codewords will be selected from F_2^{16} . It is denoted as follows:

$$\begin{aligned} \mathbf{a} &= \{\mathbf{a}_i, i = 0, 1, 2, \dots, 19\} \\ &= \{\mathbf{a}_0, \mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4, \mathbf{a}_5, \mathbf{a}_6, \mathbf{a}_7, \mathbf{a}_8, \mathbf{a}_9, \mathbf{a}_{10}, \mathbf{a}_{11}, \\ &\quad \mathbf{a}_{12}, \mathbf{a}_{13}, \mathbf{a}_{14}, \mathbf{a}_{15}, \mathbf{a}_{16}, \mathbf{a}_{17}, \mathbf{a}_{18}, \mathbf{a}_{19}\}, \end{aligned}$$

Here, the index i indicates the ordering of vector elements, not the rows or columns of a matrix. In Python, the NumPy library facilitates matrix and

algebraic operations, while the random (or `numpy.random`) module is used to select the initial codes randomly. The outcomes of this selection are summarized in the following sections. The results obtained are as follows.

$$\begin{aligned}
 \mathbf{a}_0 &= (1101111110010010) \\
 \mathbf{a}_1 &= (1001101110111000) \\
 \mathbf{a}_2 &= (1011010000010011) \\
 \mathbf{a}_3 &= (0110101101101000) \\
 \mathbf{a}_4 &= (0110000001100111) \\
 \mathbf{a}_5 &= (1101011000100101) \\
 \mathbf{a}_6 &= (1000000000010100) \\
 \mathbf{a}_7 &= (0000100010100011) \\
 \mathbf{a}_8 &= (1001001011100000) \\
 \mathbf{a}_9 &= (1101001111110010) \\
 \mathbf{a}_{10} &= (1010011110111000) \\
 \mathbf{a}_{11} &= (1000111011001010) \\
 \mathbf{a}_{12} &= (1100111101000011) \\
 \mathbf{a}_{13} &= (1000000001100110) \\
 \mathbf{a}_{14} &= (0110101001000011) \\
 \mathbf{a}_{15} &= (1011110011000111) \\
 \mathbf{a}_{16} &= (1010010011101010) \\
 \mathbf{a}_{17} &= (0100111110011110) \\
 \mathbf{a}_{18} &= (1011011001000000) \\
 \mathbf{a}_{19} &= (1001111010101010)
 \end{aligned}$$

2. Defining the generator matrix. As introduced in [23], there is a generator matrix denoted $\tau_2(v)$, hereafter referred to as the τ matrix. In this study, the τ matrix serves as the generator matrix:

$$\tau = \begin{pmatrix}
 0000000101111111 \\
 0000001010111111 \\
 0100000011011111 \\
 1000000011101111 \\
 0001000011110111 \\
 0010000011111011 \\
 0000010011111101 \\
 0000100011111110 \\
 0111111100010000 \\
 1011111100100000 \\
 1101111100000100 \\
 1110111100001000 \\
 1111011100000001 \\
 1111101100000010 \\
 1111110101000000 \\
 1111111010000000
 \end{pmatrix}$$

After determining the initial code \mathbf{a} , the resulting codeword is obtained by multiplying \mathbf{a} by τ . Let

$$\mathbf{b} = \{\mathbf{a}\tau \in F_2^{16} | \mathbf{a} \in F_2^{16}\} = \{\mathbf{a}\tau \in F_2^{16} | \mathbf{a} \in F_2^{16}\}.$$

Hence, each element \mathbf{b} in the above set is formed by applying the τ matrix to a corresponding initial code \mathbf{a} . The code word results from the τ matrix as a generator matrix:

$$\begin{aligned}
 \mathbf{b}_0 &= (1001011010100101) \\
 \mathbf{b}_1 &= (0110000110110101) \\
 \mathbf{b}_2 &= (1111001000111000) \\
 \mathbf{b}_3 &= (0111001011000110) \\
 \mathbf{b}_4 &= (1011110000001110) \\
 \mathbf{b}_5 &= (1010000001000001) \\
 \mathbf{b}_6 &= (0000010011011111) \\
 \mathbf{b}_7 &= (1010001010111011) \\
 \mathbf{b}_8 &= (0100011100101110) \\
 \mathbf{b}_9 &= (1111010111101011) \\
 \mathbf{b}_{10} &= (0110111010001001) \\
 \mathbf{b}_{11} &= (1001100110011001) \\
 \mathbf{b}_{12} &= (0100111111000001) \\
 \mathbf{b}_{13} &= (0111011000011001) \\
 \mathbf{b}_{14} &= (1101100101100100) \\
 \mathbf{b}_{15} &= (0010011101101100) \\
 \mathbf{b}_{16} &= (0000001100001100) \\
 \mathbf{b}_{17} &= (1000001000000101) \\
 \mathbf{b}_{18} &= (1110000101001011) \\
 \mathbf{b}_{19} &= (1111000100110100)
 \end{aligned}$$

This is a description of the steps in constructing a binary code from the group matrix ring $M_2(R)G$ such that the codeword \mathbf{b} for the random message is obtained. This code word will later be sent to the message recipient. This process is called encoding where the initial message/code is operated to produce a codeword.

Recommendation

This study has demonstrated how to utilize the matrix generator method, specifically the group matrix ring $M_2(R)G$, in constructing binary linear codes. For future research, several directions can be explored to enhance the efficiency and applicability of the proposed method. Alternative coding techniques, such as the H parity-check matrix method, may offer further improvements in error correction capabilities. Additionally, investigating the

integration of algebraic coding theory with modern cryptographic frameworks could contribute to advancements in secure communication systems. Furthermore, computational optimizations, including machine learning-assisted decoding and deep learning-based error detection, could be explored to improve the performance of algebraic codes in large-scale applications. Expanding the scope of this research to cover quantum error correction and network coding could also provide valuable insights into the broader implications of algebraic approaches in information security.

Conclusion

This article has presented an algebraic approach to strengthening information security by leveraging ring matrix groups as a basis for constructing robust codes. Through the systematic use of algebraic structures—particularly those derived from dihedral groups and other finite groups—researchers and practitioners can generate binary linear codes with advantageous error-correction properties and enhanced resilience to potential attacks. The inherent symmetry and rich algebraic properties of these groups offer a flexible framework for designing codes that are both computationally efficient and theoretically sound. Furthermore, by integrating group and ring theory with practical software tools (e.g., Python) for large-scale computations, the proposed methodology can be applied to real-world scenarios where manual calculations become infeasible. This combination of algebraic rigor and computational feasibility paves the way for advanced security mechanisms, including improved key management and robust data integrity checks.

Acknowledgement

This research was funded by Hibah Riset Unpad Riset Percepatan Lektor Kepala (RPLK) Universitas Padjadjaran, Indonesia, grant number 1549/UN6.3.1/PT.00/2023 and and through the Academic Leadership Grant, No. 1518/UN6.3.1/PT.00/2024 awarded to Nursanti Anggriani and supported by the Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Padjadjaran, Indonesia.

References

- [1] Koupaei, A. (2023). The Evolving Nature of Cyber Attacks and the Need for Proactive Defense Measures. *Journal of Research in Engineering and Applied Sciences*. <https://doi.org/10.46565/jreas.202384634-639>.
- [2] Shinde, S., & Ansurkar, G. (2023). Upcoming Threats in Cyber-Security. *International Journal of Scientific Research in Science and Technology*. <https://doi.org/10.32628/ijrst523102121>.
- [3] Thakuria, L., & Goswami, P. (2020). The state of cyber security: Theemerging threat trends. *The Clarion-International Multidisciplinary Journal*, 9, 61-64. <https://doi.org/10.5958/2277-937X.2020.00016.7>.
- [4] Gracy, M., Jeyavadhanam, B., Babu, P., Karthick, S., & Chandru, R. (2023). Growing Threats Of Cyber Security: Protecting Yourself In A Digital World. *2023 International Conference on Networking and Communications (ICNWC)*, 1-5. <https://doi.org/10.1109/ICNWC57852.2023.10127398>.
- [5] Cascavilla, G., Tamburri, D., & Heuvel, W. (2021). Cybercrime threat intelligence: A systematic multi-vocal literature review. *Comput. Secur.*, 105, 102258. <https://doi.org/10.1016/J.COSE.2021.102258>.
- [6] Mammadova, K., & Aslanov, R. (2023). Installation of integrated intellectual information security systems in open corporate networks – DDoS attack. *InterConf*. <https://doi.org/10.51582/interconf.19-20.04.2023.069>.
- [7] Del Pozo Durango, R., T, S., Díaz, E., Trejo, J., Gallegos, L., Arellano, M., Rosillo, V., & Hifóng, M. (2023). Analysis of Information Security Management Applying International Standards to Mitigate Risks. *2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE)*, 669-674. <https://doi.org/10.1109/CSCE60160.2023.00114>.
- [8] Figueira, P., López-Bravo, C., & López, J. (2020). Improving information security risk analysis by including threat-occurrence predictive models. *Comput. Secur.*, 88. <https://doi.org/10.1016/j.cose.2019.101609>.
- [9] Alshboul, A. Information Systems Security Measures and Countermeasures: Protecting Organizational Assets from Malicious Attacks. *Communications of the IBIMA*, (2010). <http://www.ibimapublishing.com/journals/CIBIMA/cibima.html>
- [10] Kidd, T. T., & Hiltbrand, R. K. Intrusion Detection and Information Security Audits. *Encyclopedia of Information Ethics and Security*, 411–417, (2007). <https://doi.org/10.4018/978-1-59140-987-8.CH061>:
- [11] Popa, D. and Simion, E. Enhancing security by combining biometrics and cryptography. *Proceedings of the 9th International Conference on Electronics, Computers and Artificial Intelligence, ECAI 2017, 2017-January*, 1–7, (2017). <https://doi.org/10.1109/ECAI.2017.8166461>
- [12] Arutyunov, V. V. The results of priority research in the field of information security. *Scientific and Technical Information Processing*, 43(1), 42–46, (2016). <https://doi.org/10.3103/S014768821601007X/METRICS>
- [13] Schaefer, R. F., Boche, H., Khisti, A., & Poor, H. V. Information Theoretic Security and Privacy of Information Systems, (2017). <https://doi.org/10.1017/9781316450840>
- [14] Shen, J., Liu, D., He, D., Huang, X., & Xiang, Y. (2020). Algebraic Signatures-Based Data Integrity Auditing for Efficient Data Dynamics in Cloud Computing. *IEEE Transactions on Sustainable Computing*, 5, 161-173. <https://doi.org/10.1109/TSUSC.2017.2781232>.
- [15] Zhao, Y., & Sun, H. (2020). Expand-and-Randomize: An Algebraic Approach to Secure Computation. *Entropy*, 23. <https://doi.org/10.3390/e23111461>.

- [16] Bhandari, A. (2021). Analysis of Computational Algebra for Cryptography and Coding Theory Applications. *Mathematical Statistician and Engineering Applications*. <https://doi.org/10.17762/msea.v70i1.2513>.
- [17] Letychevskiy, O., & Peschanenko, V. (2022). Applying Algebraic Virtual Machine to Cybersecurity Tasks. 2022 IEEE 9th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), 161-169. <https://doi.org/10.1109/SETIT54465.2022.9875895>.
- [18] Sakib, S. (2021). Analysis on Fundamental Algebraic Concepts and Information Security System. <https://doi.org/10.31224/osf.io/j46tg>.
- [19] Müller, R. R., Gäde, B., & Bereyhi, A. Linear computation coding. ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2021-June, 5065-5069, (2021). <https://doi.org/10.1109/ICASSP39728.2021.9414317>
- [20] Lavrauw, M., Storme, L., & Van de Voorde, G. Linear codes from projective spaces. 185-202, (2010). <https://doi.org/10.1090/CONM/523/10326>
- [21] Ding, C. A construction of binary linear codes from Boolean functions. *Discrete Mathematics*, **339**(9), 2288-2303, (2016). <https://doi.org/10.1016/J.DISC.2016.03.029>
- [22] Qian, L., Cao, X., Lu, W., & Solé, P. A new method for constructing linear codes with small hulls. *Designs, Codes, and Cryptography*, **90**(11), 2663-2682, (2022). <https://doi.org/10.1007/S10623-021-00940-1/TABLES/1>
- [23] Dougherty, S. T., Korban, A., Şahinkaya, S., & Ustun, D. Group matrix ring codes and constructions of self-dual codes. *Applicable Algebra in Engineering, Communications and Computing*, **34**(2), 279-299, (2023). <https://doi.org/10.1007/s00200-021-00504-9>
- [24] Korban, A., Şahinkaya, S., & Ustun, D. New singly and doubly even binary [72,36,12] self-dual codes from $M_2(R)G$ - group matrix rings. *Finite Fields and Their Applications*, **76**, 101924, (2021a). <https://doi.org/10.1016/J.FFA.2021.101924>



Sisilia Sylviani is a faculty member at the Department of Mathematics, Universitas Padjadjaran, Indonesia. She specializes in the fields of algebra and its application, cryptography, and multimedia computing. She has been actively

involved in international research collaborations, particularly in the areas of Applied Linear Algebra, and also engaged in interdisciplinary research focused on developing innovative and engaging methods for teaching mathematics. As a Lecturer and researcher, she has published in reputable journals and conferences. In addition to her research, she serves as a reviewer for various international journals and has contributed to the academic development of students through teaching and mentorship.

Nita Anastasya received bachelor degree from Universitas Padjadjaran. She was a student at Mathematics Study Program Universitas Padjadjaran and graduated in 2023. Her research interests is Applied Algebra.



Emma Carnia is currently a Faculty Member at the Department of Mathematics, Universitas Padjadjaran, where she has been teaching since 1991. Her research interests include algebra and its applications.



Fahmi Candra Permana is a Faculty Member at Department of Software Engineering, Universitas Pendidikan Indonesia, Kampus UPI di Cibiru, Universitas Pendidikan Indonesia since 2017. Currently, He also a doctoral student at the School of Electrical Engineering and

Informatics, Institut Teknologi Bandung, Indonesia. His research interests focus on the intersection of Multimedia Computing, Text Mining, and Sentiment Analysis, with a particular emphasis on Quality of Experience (QoE) and Quality of Service (QoS) in network performance.



Nursanti Anggriani is a professor of Applied Mathematics specializing in Biomathematics. She is a faculty member at the Department of Mathematics, Universitas Padjadjaran (Unpad), with a research focus on mathematical modeling in biology, epidemiology, and population dynamics. Her scholarly works have been published in reputable international journals.