Generalized Inverse Matrix Construction for Code Based Cryptography

Farshid, Haidary Makoui[1] and T. Aaron, Gulliver[1]

[1] *Department of Electrical and Computer Engineering, University of Victoria, Victoria, B.C., Canada. email: makoui@uvic.ca and agullive@ece.uvic.ca*

Abstract

The generalized inverses of systematic non-square binary matrices have applications in mathematics, channel coding and decoding, navigation signals, machine learning, data storage and cryptography such as the McEliece and Niederreiter public-key cryptosystems. A systematic non-square $(n-k)\times k$ matrix H, $n > k$, has $2^{k\times(n-k)}$ different generalized inverse matrices. This paper presents an algorithm for generating these matrices and compares it with two well-known methods, i.e. Gauss-Jordan elimination and Moore-Penrose methods. A random generalized inverse matrix construction method is given which has a lower execution time than the Gauss-Jordan elimination and Moore-Penrose approaches.

Keywords: Code-Based Cryptography, Generalized Inverse Binary Matrix, Error-Correcting Applications, Parity Check Inverse Matrix, Public Key Cryptosystem (PKC)

1 Introduction

The generalized inverse of a systematic binary matrix is used for decoding in all applications of error-correcting codes including digital communication [1], navigation signals [2], data storage systems [3] and coding theory [4] in cryptography. Generalized inverse matrices can be obtained using Gauss-Jordan elimination [5] and Moore-Penrose pseudoinverse (MPP) techniques [6] [7].

A matrix is invertible if it has full rank. A non-square matrix A with m rows and n columns where $n > m$ is full rank if it is a full row rank matrix, where the rows are linearly independent.

Gauss-Jordan elimination is used to solve linear systems $Ax = b$ by employing row reduction operations to transform augmented matrices $[A|b]$ to row-echelon form (REF). This technique also provides a reduced row-echelon form (RREF) where the leading coefficient in each row is the only non-zero element entry in its column. Gauss-Jordan elimination uses an augmented matrix to construct the nullspace of the matrix A [8] and its associated vectors that lead to the generalized inverse of full rank matrices.

The Moore-Penrose technique provides a single pseudoinverse matrix, where the multiplication of the matrix and its pseudoinverse approximately equal the identity matrix. The MPP can provide a pseudoinverse for any matrix. This technique is a useful tool for application with data analysis, optimization, neural network and machine learning applications [9].

Non-square binary matrices are used in error-correction coding, code-based cryptography and decoding algorithms [10] [11]. This present paper introduces an efficient algorithm for calculating all the generalized inverses of a binary matrix. A simplified algorithm is also given to construct a random generalized inverse matrix with lower processing time in comparison with Moore-Penrose and Gauss-Jordan methods.

1.1 Binary Linear Block Codes

In modern communication systems, redundant bits are added to a message sequence to detect and correct errors introduced by a noisy channel. The encoder assigns a binary codeword $\boldsymbol{c} = (c_1, c_2, ..., c_n)$ to a message $\boldsymbol{m} = (m_1, m_2, ..., m_k)$. For a k-tuple message $\boldsymbol{m}$, there are 2^k distinct messages and thus codewords. The set of all 2^k codewords is referred to a $C(n, k)$ block code. The length of a $C(n, k)$ block code is shown by n and k denoting dimension where $k \leq n$.

The channel encoder adds redundancy in the binary information sequence to the transmitted codewords, so each codeword has $n - k$ redundant bits more than the message associated with it. The message can scramble, permute and change the bits in the corresponding codeword [12]. These redundant bits are used by the channel decoder at the receiver's end to detect and correct errors having occurred over a noisy channel.

A $C(n,k)$ code is linear when its codewords form a k-dimensional vector subspace of the n-tuple vector space. Therefore, there are k linearly independent codewords $\boldsymbol{g}_1, \boldsymbol{g}_2, ..., \boldsymbol{g}_k$ that are settled as the rows of the generator matrix. The systematic form of generator matrix G in linear code is given by

$$G_{k\times n} = (I_k | P_{k\times(n-k)}), \tag{1}$$

where I_k is the $k \times k$ identity matrix and $P_{k\times(n-k)}$ is called the parity matrix. This can be written as

$$G = \left(\begin{array}{c|ccccc} & p_{1,1} & p_{1,2} & p_{1,3} & \cdots & p_{1,(n-k)} \\ & p_{2,1} & p_{2,2} & p_{2,3} & \cdots & p_{2,(n-k)} \\ I_k & p_{3,1} & p_{3,2} & p_{3,3} & \cdots & p_{3,(n-k)} \\ & \vdots & \vdots & \vdots & & \vdots \\ & p_{k,1} & p_{k,2} & p_{k,3} & \cdots & p_{k,(n-k)} \end{array}\right).$$

A parity check matrix H is an $(n-k) \times n$ matrix, such that $GH^T = \mathbf{0}$ where ^{T} denotes transpose, so H is a basis of the dual space of $C_{n,k}$. Thus, H generates the dual code $C^{\perp}(n,k)$ with 2^{n-k} codewords. This matrix can be employed to determine if a particular vector is a codeword. The H matrix can also be used for decoding algorithms [11]. A systematic parity check matrix has the form

$$H_{(n-k)\times n} = (P^T_{(n-k)\times k} | I_{n-k}). \tag{2}$$

which can be expressed as

$$H = \left(\begin{array}{ccccc|c} p_{1,1} & p_{2,1} & p_{3,1} & \cdots & p_{k,1} & \\ p_{1,2} & p_{2,2} & p_{3,2} & \cdots & p_{k,2} & \\ p_{1,3} & p_{2,3} & p_{3,3} & \cdots & p_{k,3} & I_{n-k} \\ \vdots & \vdots & \vdots & & \vdots & \\ p_{1,(n-k)} & p_{2,(n-k)} & p_{3,(n-k)} & \cdots & p_{k,(n-k)} & \end{array}\right),$$

denote the generalized inverse of this matrix as

$$H^{-1}_{n\times(n-k)} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,(n-k)} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,(n-k)} \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,(n-k)} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,(n-k)} \end{pmatrix}, \tag{3}$$

so that $H_{(n-k)\times n}H^{-1}_{n\times(n-k)} = I_{n-k}$, which can be expressed as

$$\left(\begin{array}{ccccc|c} p_{1,1} & p_{2,1} & p_{3,1} & \cdots & p_{k,1} & \\ p_{1,2} & p_{2,2} & p_{3,2} & \cdots & p_{k,2} & \\ p_{1,3} & p_{2,3} & p_{3,3} & \cdots & p_{k,3} & I_{n-k} \\ \vdots & \vdots & \vdots & & \vdots & \\ p_{1,(n-k)} & p_{2,(n-k)} & p_{3,(n-k)} & \cdots & p_{k,(n-k)} & \end{array}\right) \times \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,(n-k)} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,(n-k)} \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,(n-k)} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,(n-k)} \end{pmatrix} = I_{n-k}. \tag{4}$$

2 Generalized Inverse Matrix Construction

The matrix H^{-1} has $n-k$ columns, each of which can have 2^k different values, so the number of matrices is $2^{k\times(n-k)}$[13]. The i-th column of H^{-1} belongs to a column set Z_i which contains 2^k vectors of length n

$$Z_i = \left\{\begin{array}{ccccc} z_{1,1} & z_{1,2} & z_{1,3} & \cdots & z_{1,2^k} \\ z_{2,1} & z_{2,2} & z_{2,3} & \cdots & z_{2,2^k} \\ z_{3,1} & z_{3,2} & z_{3,3} & \cdots & z_{3,2^k} \\ \vdots & \vdots & \vdots & & \vdots \\ z_{k,1} & z_{k,2} & z_{k,3} & \cdots & z_{k,2^k} \\ --- & --- & --- & -- & --- \\ z_{(k+1),1} & z_{(k+1),2} & z_{(k+1),3} & \cdots & z_{(k+1),2^k} \\ z_{(k+2),1} & z_{(k+2),2} & z_{(k+2),3} & \cdots & z_{(k+2),2^k} \\ z_{(k+3),1} & z_{(k+3),2} & z_{(k+3),3} & \cdots & z_{(k+3),2^k} \\ \vdots & \vdots & \vdots & & \vdots \\ z_{n,1} & z_{n,2} & z_{n,3} & \cdots & z_{n,2^k} \end{array}\right\}. \tag{5}$$

This set can be divided into two subsets, Z_i^1 and Z_i^2, where Z_i^1 contains rows 1 to k and Z_i^2 contains rows $k+1$ to n, so that

$$Z_i^1 = \begin{Bmatrix} z_{1,1} & z_{1,2} & z_{1,3} & \cdots & z_{1,2^k} \\ z_{2,1} & z_{2,2} & z_{2,3} & \cdots & z_{2,2^k} \\ z_{3,1} & z_{3,2} & z_{3,3} & \cdots & z_{3,2^k} \\ \vdots & \vdots & \vdots & & \vdots \\ z_{k,1} & z_{k,2} & z_{k,3} & \cdots & z_{k,2^k} \end{Bmatrix}, \tag{6}$$

$$Z_i^2 = \begin{Bmatrix} z_{(k+1),1} & z_{(k+1),2} & z_{(k+1),3} & \cdots & z_{(k+1),2^k} \\ z_{(k+2),1} & z_{(k+2),2} & z_{(k+2),3} & \cdots & z_{(k+2),2^k} \\ z_{(k+3),1} & z_{(k+3),2} & z_{(k+3),3} & \cdots & z_{(k+3),2^k} \\ \vdots & \vdots & \vdots & & \vdots \\ z_{n,1} & z_{n,2} & z_{n,3} & \cdots & z_{n,2^k} \end{Bmatrix}, \tag{7}$$

Z_i^1 contains all 2^k possible binary vectors from all zeros to all ones. For example, if $k = 3$ then Z_i^1 contains the eight binary vectors of length 3

$$Z_i^1 = \begin{Bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{Bmatrix}.$$

For Z_i^2, the value of $z_{(k+b),d}$, $1 \le b \le n-k, 1 \le d \le 2^k$, is determined as follows. Multiplication of H by a column of Z_1 must satisfy

$$\begin{pmatrix} p_{1,1} & p_{2,1} & \cdots & p_{k,1} & | & \\ p_{1,2} & p_{2,2} & \cdots & p_{k,2} & | & I_{n-k} \\ \vdots & \vdots & & \vdots & | & \\ p_{1,(n-k)} & p_{2,(n-k)} & \cdots & p_{k,(n-k)} & | & \end{pmatrix} \times \begin{pmatrix} z_{1,d} \\ z_{2,d} \\ \vdots \\ z_{k,d} \\ --- \\ z_{(k+1),d} \\ z_{(k+2),d} \\ \vdots \\ z_{n,d} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{8}$$

Thus, for $b = 1$ the result is 1, and otherwise, it is 0.

so, if $b = 1$

$$z_{(k+1),d} = 1 + p_{1,1}z_{1,d} + p_{2,1}z_{2,d} + \cdots + p_{k,1}z_{k,d}, 1 \leq d \leq 2^k,$$

and if $b \neq 1$

$$z_{(k+b),d} = p_{1,b}z_{1,d} + p_{2,b}z_{2,d} + \cdots + p_{k,b}z_{k,d}, 1 \leq d \leq 2^k.$$

The columns of Z_2 satisfy

$$\begin{pmatrix} p_{1,1} & p_{2,1} & \cdots & p_{k,1} & | & \\ p_{1,2} & p_{2,2} & \cdots & p_{k,2} & | & I_{n-k} \\ \vdots & \vdots & \ddots & \vdots & | & \\ p_{1,(n-k)} & p_{2,(n-k)} & \cdots & p_{k,(n-k)} & | & \end{pmatrix} \times \begin{pmatrix} z_{1,d} \\ z_{2,d} \\ \vdots \\ z_{k,d} \\ --- \\ z_{(k+1),d} \\ z_{(k+2),d} \\ \vdots \\ z_{n,d} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}, \qquad (9)$$

so for $b = 2$

$$z_{(k+2),d} = 1 + p_{1,2}z_{1,d} + p_{2,2}z_{2,d} + \cdots + p_{k,2}z_{k,d}, 1 \leq d \leq 2^k,$$

and for $b \neq 2$

$$z_{(k+b),d} = p_{1,b}z_{1,d} + p_{2,b}z_{2,d} + \cdots + p_{k,b}z_{k,d}, 1 \leq d \leq 2^k.$$

Similarly, the columns of Z_{n-k} must satisfy

$$\begin{pmatrix} p_{1,1} & p_{2,1} & \cdots & p_{k,1} & | & \\ p_{1,2} & p_{2,2} & \cdots & p_{k,2} & | & I_{n-k} \\ \vdots & \vdots & \ddots & \vdots & | & \\ p_{1,(n-k)} & p_{2,(n-k)} & \cdots & p_{k,(n-k)} & | & \end{pmatrix} \times \begin{pmatrix} z_{1,d} \\ z_{2,d} \\ \vdots \\ z_{k,d} \\ --- \\ z_{(k+1),d} \\ z_{(k+2),d} \\ \vdots \\ z_{n,d} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}, \qquad (10)$$

so for $b = n - k$ the result is 1 and for $b \neq n - k$ the result is 0. Thus if $b = n - k$

$$z_{(k+(n-k)),d} = z_{n,d} = 1 + p_{1,(n-k)}z_{1,d} + p_{2,(n-k)}z_{2,d} + \cdots + p_{k,(n-k)}z_{k,d}), 1 \leq d \leq 2^k,$$

and if $b \neq n - k$

$$z_{(k+b),d} = p_{1,b}z_{1,d} + p_{2,b}z_{2,d} + \cdots + p_{k,b}z_{k,d}, 1 \leq d \leq 2^k.$$

2.1 Example

Let $n = 6$ and $k = 3$ with

$$G = (I_k|P_{k\times(n-k)}) = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix},$$

and

$$H = (P^T|I_{n-k}) = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix},$$

Thus, H^{-1} has $n - k = 3$ columns and there are three column sets Z_1, Z_2 and Z_3 available $(1 \leq i \leq n - k)$ with a total of $2^{k\times(n-k)} = 2^{3\times3} = 512$ possible matrices. The sets Z_i^1 and Z_i^2 are defined as follows. Z_i^1 is common for all i and is given by

$$Z_i^1 = \begin{Bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{Bmatrix},$$

and Z_i^2 can be expressed as

$$Z^2 = \begin{Bmatrix} z_{(k+1),1} & z_{(k+1),2} & z_{(k+1),3} & z_{(k+1),4} & z_{(k+1),5} & z_{(k+1),6} & z_{(k+1),7} & z_{(k+1),8} \\ z_{(k+2),1} & z_{(k+2),2} & z_{(k+2),3} & z_{(k+2),4} & z_{(k+2),5} & z_{(k+2),6} & z_{(k+2),7} & z_{(k+2),8} \\ z_{(k+3),1} & z_{(k+3),2} & z_{(k+3),3} & z_{(k+3),4} & z_{(k+3),5} & z_{(k+3),6} & z_{(k+3),7} & z_{(k+3),8} \end{Bmatrix}.$$

Combining Z_i^1 and Z_i^2 gives

$$Z_i = \left\{ \begin{array}{cccccccc} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ -- & -- & -- & -- & -- & -- & -- & -- \\ z_{4,1} & z_{4,2} & z_{4,3} & z_{4,4} & z_{4,5} & z_{4,6} & z_{4,7} & z_{4,8} \\ z_{5,1} & z_{5,2} & z_{5,3} & z_{5,4} & z_{5,5} & z_{5,6} & z_{5,7} & z_{5,8} \\ z_{6,1} & z_{6,2} & z_{6,3} & z_{6,4} & z_{6,5} & z_{6,6} & z_{6,7} & z_{6,8} \end{array} \right\}.$$

For $i = 1$, we have

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} 0 \\ 0 \\ 0 \\ - \\ z_{4,1} \\ z_{5,1} \\ z_{6,1} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix},$$

so

$$\begin{aligned} z_{41} &= 1 + (0)(0) + (1)(0) + (1)(0) = 1, \\ z_{51} &= (1)(0) + (1)(0) + (0)(0) = 0, \\ z_{61} &= (1)(0) + (0)(0) + (1)(0) = 0. \end{aligned}$$

The elements of Z_1^2 are

$$\begin{aligned} z_{4,d} &= 1 + p_{1,1}z_{1,d} + p_{2,1}z_{2,d} + p_{3,1}z_{3,d}, \\ z_{5,d} &= p_{1,2}z_{1,d} + p_{2,2}z_{2,d} + p_{3,2}z_{3,d}, \\ z_{6,d} &= p_{1,3}z_{1,d} + p_{2,3}z_{2,d} + p_{3,3}z_{3,d}, \end{aligned}$$

so

$$Z_1 = \left\{ \begin{array}{cccccccc} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ -- & -- & -- & -- & -- & -- & -- & -- \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{array} \right\}.$$

The elements of Z_2^2 are

$$\begin{aligned} z_{4,d} &= p_{1,1}z_{1,d} + p_{2,1}z_{2,d} + p_{3,1}z_{3,d}, \\ z_{5,d} &= 1 + p_{1,2}z_{1,d} + p_{2,2}z_{2,d} + p_{3,2}z_{3,d}, \\ z_{6,d} &= p_{1,3}z_{1,d} + p_{2,3}z_{2,d} + p_{3,3}z_{3,d}, \end{aligned}$$

so

$$Z_2 = \left\{ \begin{array}{cccccccc} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ -- & -- & -- & -- & -- & -- & -- & -- \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{array} \right\}.$$

The elements of Z_3^2 are given by

$$\begin{aligned} z_{4,d} &= p_{1,1}z_{1,d} + p_{2,1}z_{2,d} + p_{3,1}z_{3,d}, \\ z_{5,d} &= p_{1,2}z_{1,d} + p_{2,2}z_{2,d} + p_{3,2}z_{3,d}, \\ z_{6,d} &= 1 + p_{1,3}z_{1,d} + p_{2,3}z_{2,d} + p_{3,3}z_{3,d}, \end{aligned}$$

so

$$Z_3 = \left\{ \begin{array}{cccccccc} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ -- & -- & -- & -- & -- & -- & -- & -- \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{array} \right\}.$$

Selecting columns from each column set Z_1, Z_2, Z_3 in order gives $2^{k\times(n-k)} = 2^9 = 512$ H^{-1} matrices which satisfy $HH^{-1} = I_{n-k}$.

2.2 Random Generalized inverse Matrix Construction

An generalized inverse matrix H^{-1} can be divided into two parts, A_1 and A_2, where A_1 consists of rows 1 to k and A_2 consists of rows $k+1$ to n

$$H^{-1}_{n\times(n-k)} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,(n-k)} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,(n-k)} \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,(n-k)} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{k,1} & a_{k,2} & a_{k,3} & \cdots & a_{k,(n-k)} \\ --- & --- & --- & -- & --- \\ a_{(k+1),1} & a_{(k+1),2} & a_{(k+1),3} & \cdots & a_{(k+1),(n-k)} \\ a_{(k+2),1} & a_{(k+2),2} & a_{(k+2),3} & \cdots & a_{(k+2),(n-k)} \\ a_{(k+3),1} & a_{(k+3),2} & a_{(k+3),3} & \cdots & a_{(k+3),(n-k)} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,(n-k)} \end{pmatrix} = \begin{pmatrix} A_1 \\ - \\ A_2 \end{pmatrix}. \tag{11}$$

A random generalized inverse matrix H^{-1} can be constructed by selecting a random A_1 and constructing the corresponding matrix A_2. For example, if $n = 20$ and $k = 12$, then A_1 contains $n - k = 8$ random binary vectors of length 12 such as

$$A_1 = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Hence, the elements of A_2 are

$$A_2 = \begin{pmatrix} a_{(k+1),1} & a_{(k+1),2} & a_{(k+1),3} & \cdots & a_{(k+1),(n-k)} \\ a_{(k+2),1} & a_{(k+2),2} & a_{(k+2),3} & \cdots & a_{(k+2),(n-k)} \\ a_{(k+3),1} & a_{(k+3),2} & a_{(k+3),3} & \cdots & a_{(k+3),(n-k)} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,(n-k)} \end{pmatrix}, \tag{12}$$

where

$$a_{(k+b),d} = \sum_{i=1}^{k} p_{ib} a_{id}, (b \neq d),$$

and

$$a_{(k+b),d} = 1 + \sum_{i=1}^{k} p_{ib} a_{id}, (b = d).$$

In general, this can be expressed as

$$a_{(k+b),d} = 2^{|b-d|} \bmod 2 + \sum_{i=1}^{k} p_{ib} a_{id}. \tag{13}$$

For example, $a_{(k+1),1}$ in A_2 is given by

$$a_{(k+1),1} = 1 + p_{11}a_{11} + p_{21}a_{21} + \cdots + p_{k1}a_{k1}.$$

The result in matrix form to construct A_2 is shown as follows.

Let $B_1 = P^T_{(n-k)\times k}$ and $B_2 = I_{n-k}$, so

$$HH^{-1} = \begin{pmatrix} B_1 | B_2 \end{pmatrix} \times \begin{pmatrix} A_1 \\ - \\ A_2 \end{pmatrix} = I_{n-k},$$

$$= B_1A_1 + B_2A_2 = I_{n-k},$$

$$A_2 = B_1A_1 + I_{n-k}, \tag{14}$$

so $A_2 = B_1A_1 + I_{n-k}$ and then

$$HH^{-1} = \begin{pmatrix} B_1 | B_2 \end{pmatrix} \times \begin{pmatrix} A_1 \\ - \\ A_2 \end{pmatrix} = \begin{pmatrix} B_1 | B_2 \end{pmatrix} \times \begin{pmatrix} A_1 \\ \hline B_1A_1 + I_{n-k} \end{pmatrix},$$

$$= B_1A_1 + B_2(B_1A_1 + I_{n-k}) = B_1A_1 + B_1A_1 + I_{n-k} = I_{n-k}.$$

The next section provides the analysis of the proposed algorithm for constructing a random generalized inverse matrix.

2.3 Construction Comparison and Analysis

In this section, the processing time of Moore-Penrose pseudoinverses and the proposed method for constructing random generalized inverse matrices are compared.
The computation time is given in Table 1 for several parameter values. As an example, the processing time required to construct the random generalized inverse of H matrix with 524×1568 would be 594 millisecond using the proposed method, compared with 2172 milliseconds using the Moore-Penrose pseudoinverse.

Matrix size	Moore-Penrose (ms)	Proposed (ms)
$k = 213$, $n = 500$	94	16
$k = 524$, $n = 1568$	2172	594
$k = 768$, $n = 2048$	5109	2368
$k = 1024$, $n = 2896$	14735	5211

Table 1: Processing time

An algorithm's computational efficiency depends on the number of arithmetic operations, algorithm complexity and the amount of resources, including time and memory, needed to run the algorithm.
Solving a system of n equations with n variables using Gauss-Jordan row elimination requires approximately $(2n^3 + 3n^2 - 5n)/3$ arithmetic operations to achieve the row echelon form (REF) [14], and $(n^3 + 3/2n^2 - 5/2n)$ arithmetic operations to form RREF which is about fifty percent more than the number of REF arithmetic operations. Hence, the number of arithmetic operations that Gauss-Jordan elimination required to form RREF for a parity check matrix H with $(n-k) \times n$ index would be $(n-k)^3 + 3/2(n-k)^2 - 5/2(n-k)$.

After performing RREF, Gauss-Jordan needs to solve a system of linear equations using the null-space approach to find the set of associated vectors. Therefore, not all the augmented matrices can form RREF, known as inconsistent matrices. When RREF is formed, additional $n(n-k-1)$ arithmetic operations need to construct a generalized inverse matrix.

There are many different choices of row combinations to perform Gauss-Jordan row elimination on large-size matrices, and finding an optimum choice of linear combinations is NP-hard [15]. In fact, there are numerous different execution sequences and

therefore time complexity is exponential [15].
Moore-Penrose requires $(n-k)^2(2n-1)$ arithmetic operations to construct a full-rank HH^T and approximately $(n-k)(2n^2-2nk-n)$ arithmetic operations, exclude determinant, to construct $H^T[HH^T]^{-1}$ of a parity check matrix H. The algorithm is less complex than Gauss-Jordan, and in fact, it is faster than the Guass-Jordan elimination algorithm.

The number of arithmetic operations the proposed method requires to construct a random generalized inverse would equal the number of operations to build $A_2 = B_1A_1+I_{n-k}$, which would be $(2k-1)(n-k)^2+(n-k)$. Therefore, the multiplication of B_1 with index $(n-k)\times k$ and A_1 with index $k\times(n-k)$ required $(2k-1)(n-k)^2$ number of arithmetic operations.

The arithmetic computation is given in Table 2 for Gauss-Jordan elimination, Moore-Penrose, and the proposed algorithm for constructing a random non-square binary generalized inverse matrix. The introduced method provides optimum choices to construct a random generalized inverse matrix with less processing time and complexity than Moore-Penrose and Gauss-Jordan elimination methods.

Gauss-Jordan Elimination	Moore-Penrose	Proposed
$(n-k)^3+3/2(n-k)^2-5/2(n-k)+n(n-k-1)$	$(n-k)^2(4n-1)-n(n-k)$	$(2k-1)(n-k)^2+(n-k)$

Table 2: Computational Cost

2.4 Key change interval comparison

Based on the security key management, it is recommended to increase the system security by changing the keys in shorter time intervals. Every time that a new key is selected, the generator matrix and its associated parity-check matrix will be replaced, the Gauss-Jordan elimination method ought to transform the H matrix to RREF and find out the associated vectors to construct a random generalized inverse matrix. For instance, finding the optimum choice of linear combinations of an H matrix with 1280 rows ($n = 2048, k = 768$) to form RREF is time-consuming and may affect the performance of the system applications. The Moore-Penrose pseudoinverse also is slower than the proposed method. In fact, any time matrix H changes, the proposed algorithm can construct a random generalized inverse matrix with less complexity and lower processing time. This fact could make the proposed algorithm a suitable candidate for any system that requires changing the key (including the code-based public key with G and H matrices) periodically in a shorter time interval.

3 Conclusion

This paper considered the construction of all H generalized inverse matrices of a non-square ($n \neq k$) matrix H. The matrix H^{-1} has $n-k$ columns. The paper proposes a column set Z_i where $1 \leq i \leq n-k$. The "i" column of H^{-1} belongs to a column set Z_i that contains 2^k vectors. It also divides the column set Z_i into two subsets which simplifies the calculation of all 2^k vectors and leads to the construction of all the $2^{k\times(n-k)}$ generalized inverse matrices.

Furthermore, the random generalized inverse matrix construction method presented, introduces matrix A_1 and A_2, where A_1 consists of $n-k$ binary vectors. In simple term, the elements of the matrix A_1 can be selected on a random basis and the matrix A_2 can be constructed using a simplified proposed equation $A_2 = B_1 A_1 + I_{n-k}$. In fact, the proposed approach provides a shorter processing time to construct a random generalized inverse matrix that can be suitable for applications that demand new keys to be generated periodically in shorter interval times.

Considering the restricted applicability of Moore-Penrose and Gauss-Jordan methods, the approach introduced in the present paper may have superiority over previous methods regarding computational simplicity and generality. In fact, it offers a shorter processing time, and computational simplicity and might be a suitable approach to providing better performance if a system demands changing or generating the keys periodically for any reason including enhancing the system security.

References

[1] C. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.,*, vol. 27, no. 3, pp. 379-423, 1948.

[2] R. Acharya, "Undrestanding satellite navigation," *Mobile geographic info systems*, Academic press, electronic books, 2014.

[3] Alexander Thomasian, "Storage Systems: Organization, Performance, Coding, Reliability, and Their Data Processing," *Storage systems book*, publisher Waltham, Massachusetts, Elsevier, 2011.

[4] S. Saraf, S. Dhingra, and G. Pinheiro, "Parallel algorithm for finding inverse of a matrix and its application in message sharing (coding theory)," *International Journal of Computer Applications*, vol. 975, p. 8887, 2016.

[5] P. S. Stanimirović and M. D. Petković, "Gauss–Jordan elimination method for computing outer inverses," *Applied Mathematics and Computation*, vol. 219, no. 9, pp. 4667–4679, Jan. 2013.

[6] J.C.A. Barata, M.S. Hussein, "The Moore-Penrose Pseudoinverse. A Tutorial Review of the Theory," *Instituto de F´ısica, Universidade de S˜ao Paulo*, C.P. 66318, 05314-970 S˜ao Paulo, SP, Brazil, Oct. 2011.

[7] H. Chen and Y. Wang, "A family of higher-order convergent iterative methods for computing the Moore–Penrose inverse," *Applied Mathematics and Computation*, vol. 218, no. 8, pp. 4012–4016, Dec. 2011.

[8] N. Guglielmi, M. L. Overton, and G. Stewart, "An efficient algorithm for computing the generalized null space decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 1, pp. 38–54, 2015.

[9] J Tapson, A van Schaik, "Learning the inverse solution to network weights," *Neural networks*, vol. 45, pp. 94–100, 2013.

[10] R. J. McEliece, "A public-key cryptosystem based on algebraic coding theory," *Jet Propulsion Lab*, DSN Tech. Rep. 42.44, pp. 114–116, 1978.

[11] H. Niederreiter, "Knapsack-type cryptosystems and algebraic coding theory," *Problems of Control and Information Theory*, vol. 15, pp. 159–166, 1986.

[12] M. Esmaeili, T.A.Gulliver, "Joint channel coding-cryptography based on random insertions and deletions in quasi-cyclic-low-density parity check codes," *IET communications*, vol.9 (12), pp. 1555-1560, 2015.

[13] M. Esmaeili, T.A.Gulliver, "Application of Linear Block Codes in Cryptography," *University of Victoria department of Electrical and Computer Engineering*, Chapter 5, Security analysis. pp. 45-53, 2019.

[14] Farebrother, R.W., "Linear least squares computations," *Statistics, textbooks and monographs*, London. Taylor and Francis. pp. 12, 2017.

[15] Fang Xin, Havas George, "On the worst-case complexity of integer Gaussian elimination," *International Conference on Symbolic and Algebraic Computation*, ISSAC '97, Proceedings of the 1997 international symposium on Symbolic and algebraic computation. pp. 28-31, July 1997.