

# REDOG

---

Ji-Hoon Hong

February 28, 2024

Department of Mathematics

Sogang University, S. Korea

## REDOG

---

1. Warm-up
2. Coding Theory
3. Code-based Cryptography
4. REDOG

# 1. Warm-up

# 1. Warm-up

## ● Operation

- A function that takes zero or more input values  
to a well-defined output value

- ex) Addition(Plus,+) in real number  $\mathbb{R}$

$$\begin{array}{rcl} 1 + 2 & = & 3 \\ \pi + \frac{3}{7} & = & \pi + \frac{3}{7} \end{array}$$

# 1. Warm-up

## ● Operation (ex: $+$ in $\mathbb{R}$ ) - binary operation

### ■ Take any $r \in \mathbb{R}$

#### - Identity

An element that makes the result is the same as the given number

$$r + 0 = 0 + r = r$$

#### - Inverse

An element that makes the result as the identity of same operation

$$r + (-r) = (-r) + r = 0$$

# 1. Warm-up

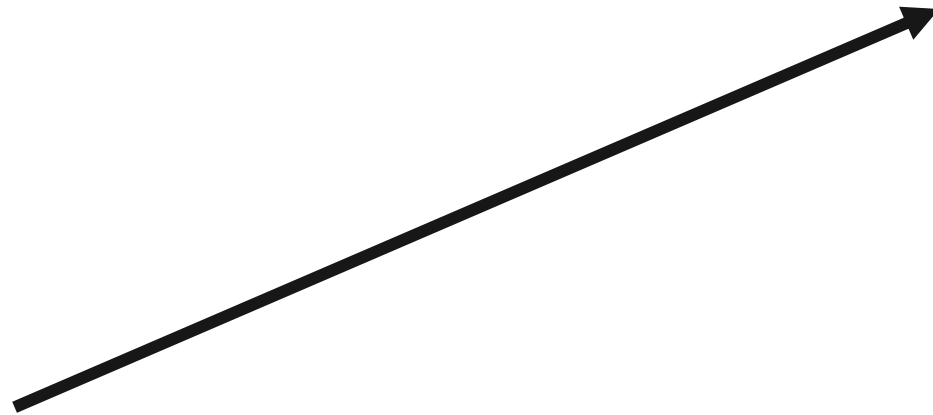
## ●Scalar

- An element of a field which is used to define a **vector space**
- ex) Real numbers (elements of a **field**)
- A variable described by a single number

# 1. Warm-up

## ● Vector

- A physical quantity that has magnitude and direction



# 1. Warm-up

## ● Vector

- ~~A physical quantity that has magnitude and direction~~
- Some quantities that a single number cannot express
- Elements of some **vector spaces**



# 1. Warm-up

- Vector

- ex) Column of numbers

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

# 1. Warm-up

## ● Vector space

- A vector space consists of a set  $V$ , a **field**  $\mathbb{F}$ , and two operations  
operation: vector addition, scalar multiplication  
with satisfy the following axioms(conditions)

- Let  $V$  be a vector space over a field  $K$ 
  - Elements of  $V$  is called vector
  - Elements of  $K$  is called scalar

# 1. Warm-up

## ● Vector space

- Vector addition (let  $\mathbf{u}, \mathbf{v}, \mathbf{w} \in V$  and  $a, b \in K$ )

- Associativity

$$\mathbf{u} + (\mathbf{v} + \mathbf{w}) = (\mathbf{u} + \mathbf{v}) + \mathbf{w}$$

- Commutativity

$$\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$$

- Identity element

$$\exists \mathbf{0} \in V \text{ s.t. } \mathbf{v} + \mathbf{0} = \mathbf{0} + \mathbf{v} = \mathbf{v}$$

- Inverse elements

$$\forall \mathbf{v} \in V, \exists -\mathbf{v} \in V \text{ s.t. } \mathbf{v} + (-\mathbf{v}) = (-\mathbf{v}) + \mathbf{v} = \mathbf{0}$$

# 1. Warm-up

## ● Vector space

- Scalar multiplication (let  $\mathbf{u}, \mathbf{v}, \mathbf{w} \in V$  and  $a, b \in K$ )

- Compatibility

$$a(b\mathbf{v}) = (ab)\mathbf{v}$$

- Identity element

$$\exists 1 \in \mathbb{F} \text{ s.t. } 1\mathbf{v} = \mathbf{v}$$

- Distributivity w.r.t. vector addition

$$a(\mathbf{u} + \mathbf{v}) = a\mathbf{u} + a\mathbf{v}$$

- Distributivity w.r.t field addition (scalar addition)

$$(a + b)\mathbf{v} = a\mathbf{v} + b\mathbf{v}$$

# 1. Warm-up

## ● Linear combination

- An expression constructed from a set of terms by multiplying each term by a constant and adding the results

- Linear combination of  $x$  and  $y$  with constant  $a$  and  $b$ :

$$ax + by$$

- Let  $V$  be a vector space over a field  $K$ . Let  $\mathbf{v}_i \in V$  and  $a_i \in K$  for  $i = 1, \dots, n$ .
- Then

$$\sum_{i=1}^n a_i \mathbf{v}_i = a_1 \mathbf{v}_1 + a_2 \mathbf{v}_2 + \cdots + a_n \mathbf{v}_n$$

# 1. Warm-up

## ● Group $(G, \circ)$

- A set with an binary operation (Take any  $g_1, g_2, g_3 \in G$ ,)

- The operation is associative

$$(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$$

- $G$  has an identity and inverse

$$\exists e \in G \text{ s.t. } e \circ g_1 = g_1 \circ e = g_1$$

$$\exists g_1^{-1} \in G \text{ s.t. } g_1 \circ g_1^{-1} = g_1^{-1} \circ g = e$$

# 1. Warm-up

## ● Group $(G, \circ)$

- A commutative group called abelian group (Take any  $g_1, g_2 \in G$ ,)

- The operation is commutative means

$$g_1 \circ g_2 = g_2 \circ g_1$$

- ex) The integer set  $\mathbb{Z}$

$(\mathbb{Z}, +)$  is a group, especially abelian group

$(\mathbb{Z}, \times)$  is not a group

# 1. Warm-up

## ● Ring

- A set( $R$ ) with two binary operations  $+$ (addition) and  $\cdot$ (multiplication) satisfying the ring axioms

- $R$  is an abelian group under addition

- $R$  is a monoid under multiplication

The multiplication is associative, and  $R$  has a multiplicative identity

- Multiplication is distributive w.r.t addition

$$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$$

$$(b + c) \cdot a = (b \cdot a) + (c \cdot a)$$



# 1. Warm-up

## ●Field

- A set  $(F)$  is a ring and  $(F, +)$  and  $(F, \cdot)$  are both abelian group
  - Identity, inverse, associativity, commutativity, and distributivity

### Example)

- $(\mathbb{Z}, +, \cdot)$  is a commutative ring, but not a field
  - It's because  $(\mathbb{Z}, \cdot)$  has no (multiplicative) inverse
- $(\mathbb{Q}, +, \cdot)$  is a field

# 1. Warm-up

## ●Matrix!

- A rectangular array of numbers, called the entries of the matrix
  - Rectangular display of vectors in rows and columns
  - Vector is just a  $n \times 1$  matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

- $Mat_{n \times m}(R)$  as for every  $n \times m$  matrices over  $R$  where  $R$  is a ring
- There are other notations such as  $GL_n(R)$ ,  $SL_n(R)$

# 1. Warm-up

## ●Matrix!

- Matrix addition(+), multiplication( $\times$ ), and scalar multiplication( $\cdot$ )
  - Let  $A, B \in \text{Mat}_{2 \times 2}(\mathbb{R})$  and  $r \in \mathbb{R}$ , such that

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

- $A + B = \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} \\ a_{21} + b_{21} & a_{22} + b_{22} \end{bmatrix}$
- $A \times B = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{bmatrix}$
- $r \cdot A = \begin{bmatrix} ra_{11} & ra_{12} \\ ra_{21} & ra_{22} \end{bmatrix}$

# 1. Warm-up

## ●Matrix!

- Is  $(\text{Mat}_{2 \times 2}(\mathbb{R}), +)$  a group?  $\rightarrow$  Yes!
- Is  $(\text{Mat}_{2 \times 2}(\mathbb{R}), \times)$  a group?  $\rightarrow$  No!
- Is  $(\text{Mat}_{2 \times 2}(\mathbb{R}), +, \times)$  a ring?  $\rightarrow$  Yes!
- Is  $(\text{Mat}_{2 \times 2}(\mathbb{R}), +, \cdot)$  a vector space?  $\rightarrow$  Yes!

## 2. Coding Theory

## 2. Coding Theory

### ● Father of Information Theory

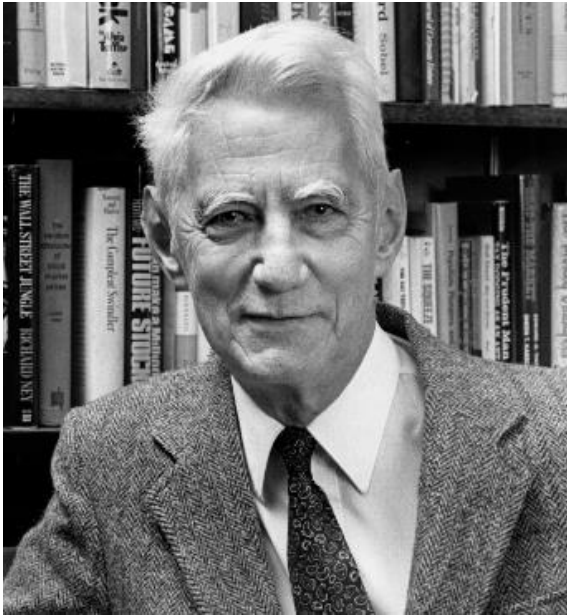
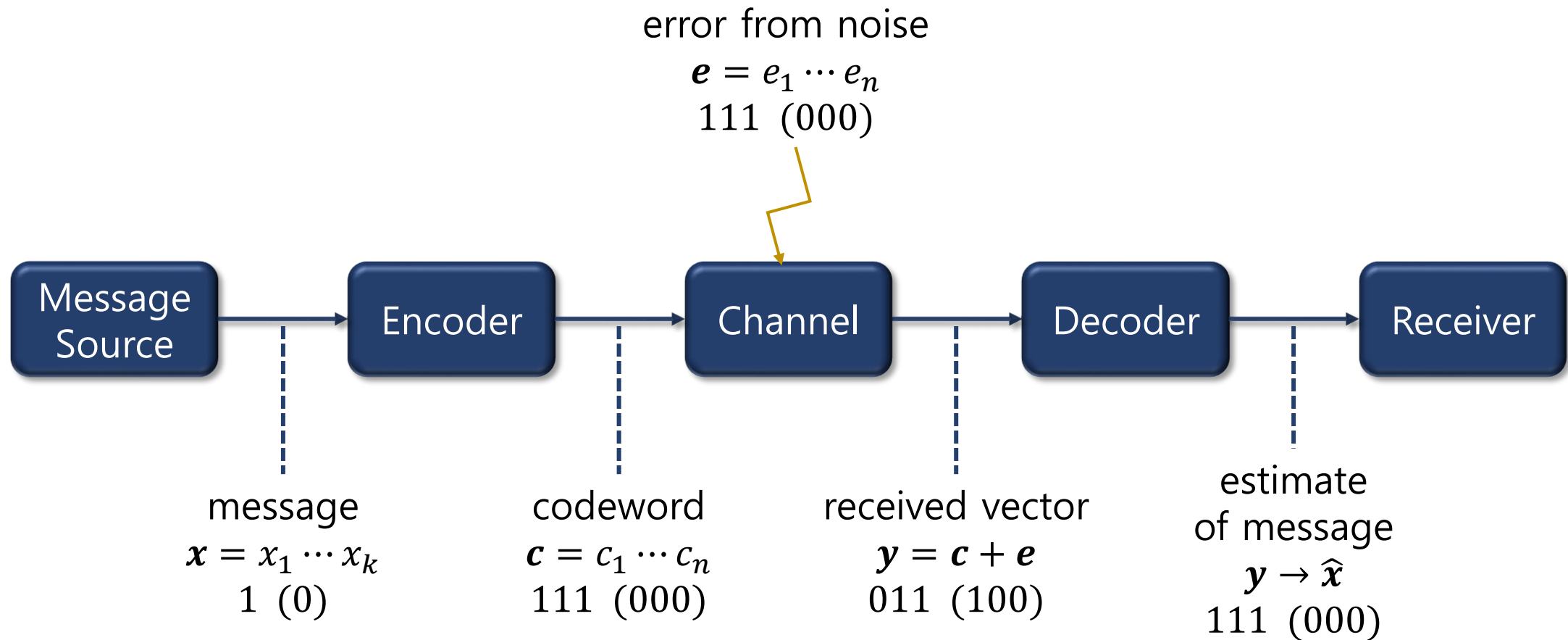


Figure: Claude Shannon  
(1916-2001)

- Shannon's paper
  - “**A Mathematical Theory of Communication**”  
on Information Theory(1948)
- Information Theory
  - Mathematical study of the quantification, storage, and communication of information
  - Information: An abstract concept that has the power to inform

## 2. Coding Theory

### ● Communication Channel



## 2. Coding Theory

### ● Father of Coding Theory



Figure: Richard Hamming  
(1915-1998)

- Discover Hamming code
  - Hamming code is an error-correcting code
- Generator matrix  $G$  of Hamming  $[7,4,3]$  code

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$



## 2. Coding Theory

### ● What is a code?

- Let  $A$  be a finite alphabet (Usually,  $A = \mathbb{Z}_2, \mathbb{Z}_p$ . other rings also available)
- $A^n = \{(x_1, \dots, x_n) | x_i \in A\}$
- An **(error-correcting) code**  $C$  over  $A$  is a subset of  $A^n$
- Elements of  $C$  are called **codewords**
- A code over  $\mathbb{Z}_2$  is called a **binary code**
- The **weight** of  $\mathbf{x} = (x_1, \dots, x_n)$  is the number of nonzero coordinates, denoted by  $\text{wt}(\mathbf{x})$ .
- The **Hamming distance**  $d(\mathbf{x}, \mathbf{y})$  between  $\mathbf{x}, \mathbf{y} \in A^n$  is  $\text{wt}(\mathbf{x} - \mathbf{y})$ 
  - Ex) if  $\mathbf{x} = (1, 0, 0, 1, 0)$  and  $\mathbf{y} = (0, 0, 1, 0, 0)$ , then  $d(\mathbf{x}, \mathbf{y}) = 3$

## 2. Coding Theory

### ● Linear codes

- A **linear code  $C$  of length  $n$  and dimension  $k$  over  $\mathbb{Z}_p$**   
:= a  $k$ -dimensional subspace of  $\mathbb{Z}_p^n$
- We denote  $C$  by an  $[n, k]$  linear code over  $\mathbb{Z}_p$
- If  $C$  is an  $[n, k]$  linear code, represent it as a  $k \times n$  generator matrix  $G$  whose rows form a basis for  $C$  and a  $(n - k) \times n$  parity check matrix  $H$  such that  $HG^T = 0$
- The minimum distance(weight)  $d$  of a linear code  $C$   
:= the minimum of  $\text{wt}(\mathbf{x})$ ,  $\mathbf{x} \neq \mathbf{0} \in C$
- Denote it by an  $[n, k, d]$  code.  $d$  can be at most  $n - k + 1$  (**Singleton bound**)
- A set of  $k$  columns of an  $[n, k, d]$  code is called an **information set** if it is linearly independent.

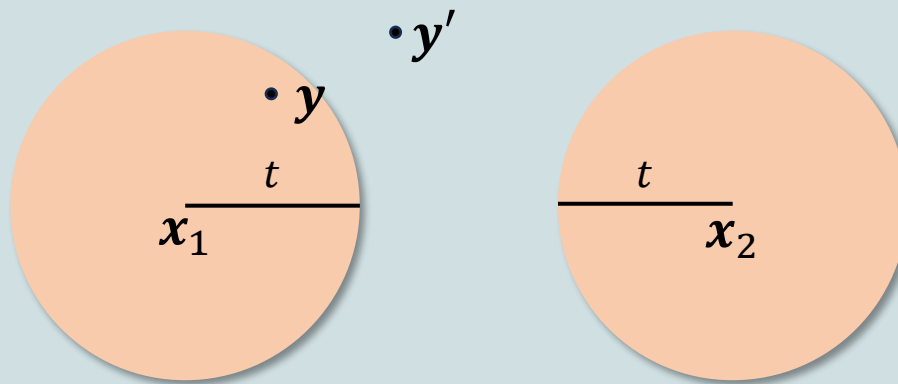
## 2. Coding Theory

### ● How many errors can correct?

#### Theorem

Any  $[n, k, d]$  linear code can correct up to  $t = \lfloor \frac{d-1}{2} \rfloor$  errors  
(by the nearest neighbor decoding)

#### Sketch of a geometric proof



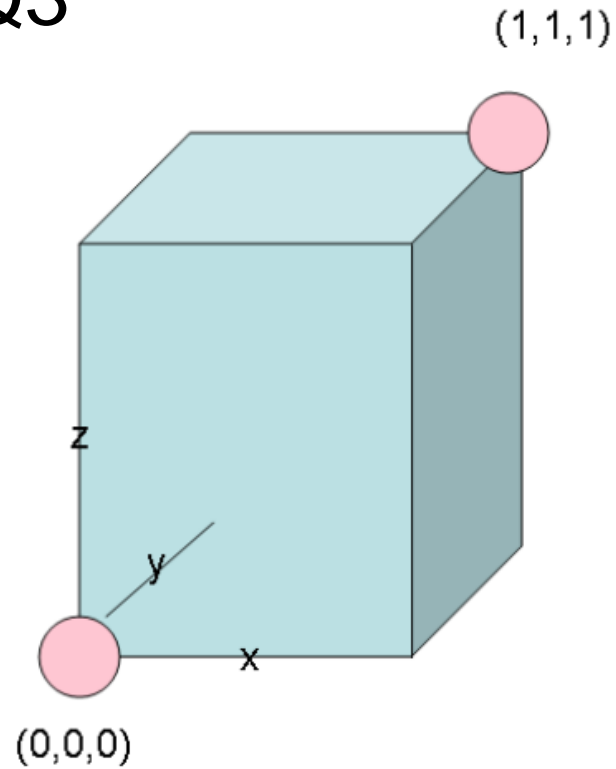
- $x_1, x_2$ : codeword
- $y, y'$ : received vector
- $2t \leq d$

1.  $y$  is uniquely decoded as  $x_1$
2.  $y'$  is not decoded

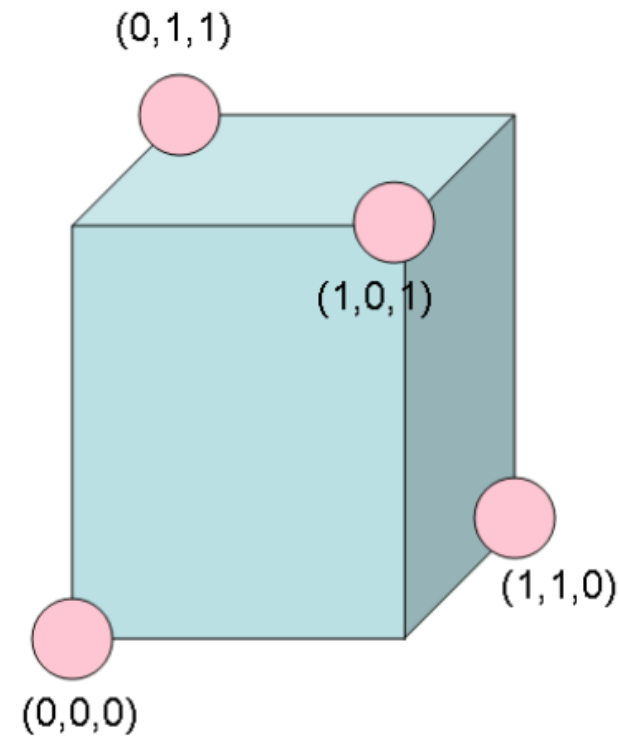
## 2. Coding Theory

- Simple examples

- Cube Q3



$$C1 = \{(0,0,0), (1,1,1)\}$$



$$C2 = \{(0,0,0), (1,1,0), (1,0,1), (0,1,1)\}$$

## 2. Coding Theory

### ●Hamming [7,4,3] code

- Generator matrix  $G$  and parity check matrix  $H$  of Hamming [7,4,3] code

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}, \quad H = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

- You can check that  $HG^T = 0$
- You can check that every codeword has a weight of at least 3

## 2. Coding Theory

### ●Hamming [7,4,3] code

#### ▪ Error correcting?

- Let  $x = (1,0,1,0)$  be a message

- Compute a codeword  $c = xG$

$$\checkmark c = (1,0,1,0,1,0,1)$$

- Add error vector  $e = (0,0,0,1,0,0,0)$

Hamming code can correct  $\left\lfloor \frac{3-1}{2} \right\rfloor = 1$  error!

- Received vector  $y = c + e = (1,0,1,1,1,0,1)$

## 2. Coding Theory

### ●Hamming [7,4,3] code

- Error correcting?

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \rightarrow$$

$x$	$c$
(0,0,0,0)	(0,0,0,0,0,0,0)
(0,0,0,1)	(0,0,0,1,1,1,1)
(0,0,1,0)	(0,0,1,0,0,1,1)
(0,1,0,0)	(0,1,0,0,1,0,1)
(1,0,0,0)	(1,0,0,0,1,1,0)
$\vdots$	$\vdots$
(1,0,1,0)	(1,0,0,0,1,0,1)
$\vdots$	$\vdots$

## 2. Coding Theory

### ●Hamming [7,4,3] code

$x$	$c$	$y = c + e = (1,0,1,1,1,0,1)$
(0,0,0,0)	(0,0,0,0,0,0,0)	3
(0,0,0,1)	(0,0,0,1,1,1,1)	3
(0,0,1,0)	(0,0,1,0,0,1,1)	4
(0,1,0,0)	(0,1,0,0,1,0,1)	4
(1,0,0,0)	(1,0,0,0,1,1,0)	3
(0,0,1,1)	(0,0,1,1,1,0,0)	3
(0,1,0,1)	(0,1,0,1,0,1,0)	6
(0,1,1,0)	(0,1,1,0,1,1,0)	5
(1,0,0,1)	(1,0,0,1,0,0,1)	3
(1,0,1,0)	(1,0,1,0,1,0,1)	1
(1,1,0,0)	(1,1,0,0,0,1,1)	6
(0,1,1,1)	(0,1,1,1,0,0,1)	3
(1,0,1,1)	(1,0,1,1,0,1,0)	3
(1,1,0,1)	(1,1,0,1,1,0,0)	3
(1,1,1,0)	(1,1,1,0,0,0,0)	4
(1,1,1,1)	(1,1,1,1,1,1,1)	2



## 2. Coding Theory

### ●Hamming [7,4,3] code

#### ▪ Error correcting?

- Let  $x = (1,0,1,0)$  be a message

- Compute a codeword  $c = xG$

$$\checkmark c = (1,0,1,0,1,0,1)$$

- Add error vector  $e = (0,0,0,1,0,0,0)$

Hamming code can correct  $\left\lfloor \frac{3-1}{2} \right\rfloor = 1$  error!

- Received vector  $y = c + e = (1,0,1,1,1,0,1)$

- Compute  $Hy^T = (1,1,1)^T$

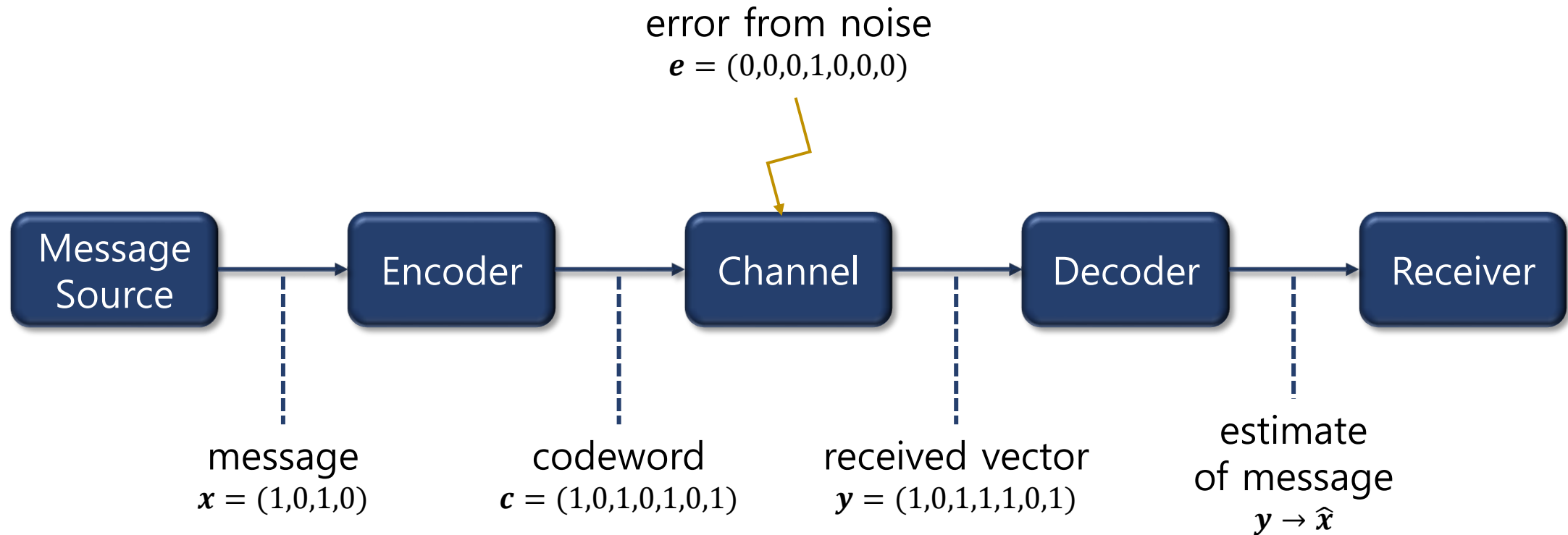
The result  $(1,1,1)^T$  is called the syndrome

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

- Find error vector  $e$  is  $e = (0,0,0,1,0,0,0)!$

## 2. Coding Theory

### ● Communication Channel



## 2. Coding Theory

### ●Decoding!

- The process of translating received messages into codewords
  - Received vector  $y = c + e = (1,0,1,1,1,0,1)$
  - Error vector  $e$  is  $e = (0,0,0,1,0,0,0)$ , so  $c = y - e = (1,0,1,0,1,0,1)$

$x$	$c$
(0,0,0,0)	(0,0,0,0,0,0,0)
(0,0,0,1)	(0,0,0,1,1,1,1)
(0,0,1,0)	(0,0,1,0,0,1,1)
(0,1,0,0)	(0,1,0,0,1,0,1)
(1,0,0,0)	(1,0,0,0,1,1,0)
(0,0,1,1)	(0,0,1,1,1,0,0)
(0,1,0,1)	(0,1,0,1,0,1,0)
(0,1,1,0)	(0,1,1,0,1,1,0)
(1,0,0,1)	(1,0,0,1,0,0,1)
(1,0,1,0)	(1,0,1,0,1,0,1)
(1,1,0,0)	(1,1,0,0,0,1,1)

We can find the message  $x$  from ciphertext  $c$

This process is called **syndrome decoding**

Others...(ISD, Maximum likelihood decoding.. etc.)

### 3. Code-based Cryptography

### 3. Code-based Cryptography

#### ●Quantum Computer

- It can use an exponentially faster algorithm than a classical computer
  - So existing crypto algorithms can be broken (RSA, ECC)
  - The existing algorithms can be solved within the polynomial time
  - To be safe from a quantum computer, the difficulty level of an algorithm should be NP-hard or higher
    - Syndrome decoding problem is NP-complete
- [Berlekamp, McEliece & van Tilborg, 1978]

### 3. Code-based Cryptography

#### ● The McEliece cryptosystem – key generation

- Select  $[n, k]$ -linear code  $C$  which can correct  $t$  errors (ex: binary Goppa code)
- Let  $G$  be a generator matrix for  $C$
- Select a random  $k \times k$  binary non-singular(invertible) matrix  $S$  and  $n \times n$  binary permutation matrix  $P$
- Compute the  $k \times n$  matrix  $\hat{G} = SG P$
- The public key is  $(\hat{G}, t)$  and the private key is  $(S, P, A)$  where  $A$  is an efficient decoding algorithm of code  $C$

### 3. Code-based Cryptography

- The McEliece cryptosystem – encryption

- Generate a message  $m$  of length  $k$
- Compute the vector  $c' = m\hat{G}$
- Generate a random  $n$ -bit error  $e$  with  $\text{wt}(e) = t$
- Compute the ciphertext  $c = c' + e$

### 3. Code-based Cryptography

#### ● The McEliece cryptosystem – decryption

- Compute the inverse of  $P$  (i.e.  $P^{-1}$ )
- Compute the vector  $\hat{c} = cP^{-1}$   
$$= m\hat{G}P^{-1} + eP^{-1} = mSG + eP^{-1}$$
- Use decoding algorithm  $A$  to decode  $\hat{c}$  to  $\hat{m}$  ( $\hat{m} = mS$ )
- Compute  $m = \hat{m}S^{-1}$



### 3. Code-based Cryptography

#### ● The Niederreiter cryptosystem – key generation

- Select a binary  $[n, k]$ -linear Goppa code  $C$  which can correct  $t$  errors
- Let  $H$  be a parity check matrix for  $C$
- Select a random  $(n - k) \times (n - k)$  binary non-singular(invertible) matrix  $S$  and  $n \times n$  binary permutation matrix  $P$
- Compute the  $(n - k) \times n$  matrix  $\hat{H} = SHP$
- The public key is  $(\hat{H}, t)$  and private key is  $(S, H, P, A)$  where  $A$  is an efficient decoding algorithm of code  $C$

### 3. Code-based Cryptography

- The Niederreiter cryptosystem – encryption and decryption

- Generate an  $n$ -bit message  $e$  with  $\text{wt}(e) = t$
- Compute the ciphertext  $c = \hat{H}e$
- Find a  $n$ -bit error  $e$  with  $\text{wt}(e) = t$  satisfying  $c = \hat{H}e$

### 3. Code-based Cryptography

#### ● McEliece vs Niederreiter

- Both use the non-singular matrix and permutation matrix to scramble the generator matrix and the parity check matrix
- The McEliece cryptosystem uses a generator matrix
- The Niederreiter cryptosystem uses a parity check matrix
- For McEliece, an attacker is faced with decoding  $c$  to nearest codeword  $m\hat{G}$
- For Niederreiter, an attacker is facing  $t$ -error correcting problem for  $\hat{H}$

# 3. Code-based Cryptography

## ●NIST PQC-competition

- 2016: NIST began the PQC standardization process
- 2017~2019: 69 algorithms were selected for Round 1
- 2019~2020: based on the evaluation and feedback from Round 1, 26 algorithms were selected for Round 2
- 2020~2022: 4 algorithms were selected for Round 3  
(CRYSTALS-Kyber, CRYSTALS-Dilithium, FALCON, and SPHINCS+)
- ~2024: select 4 more algorithms(BIKE, Classic McEliece, HQC, and SIKE)

# 3. Code-based Cryptography

## ●McNie

- J.-L. Kim et al. suggest the McNie at NIST PQC competition
- It was selected for Round 1
- McNie uses both generator matrix and parity check matrix
- McNie uses **rank-metric code** for efficiency (LRPC code)

### 3. Code-based Cryptography

#### ● Rank metric codes

- Let be  $\{a_1, a_2, \dots, a_m\}$  a basis of  $\mathbb{F}_{q^m}$  over  $\mathbb{F}_q$
- Write  $x \in \mathbb{F}_{q^m}$  as  $x = \sum_{i=1}^m X_i a_i$ , where  $X_i \in \mathbb{F}_q$
- Extend to  $\mathbf{v} = (v_1, \dots, v_n) \in \mathbb{F}_{q^m}^n$  as the map  $Mat: \mathbb{F}_{q^m}^n \rightarrow \mathbb{F}_q^{m \times n}$  defined by

$$\mathbf{v} \mapsto \begin{bmatrix} V_{11}, & V_{21} & \dots & V_{n1} \\ V_{12}, & V_{22} & \dots & V_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ V_{1m}, & V_{2m} & \dots & V_{nm} \end{bmatrix}$$

### 3. Code-based Cryptography

#### ● Rank metric codes

- The rank weight of  $\mathbf{v}$  is then  $\text{wt}(\mathbf{v}) := \text{rk}_q(\text{Mat}(\mathbf{v}))$
- The rank distance between  $\mathbf{v}, \mathbf{w} \in \mathbb{F}_{q^m}^n$  is  $d_R := \text{wt}_R(\mathbf{v} - \mathbf{w})$
- A rank metric  $[n, k, d]$  code  $C$  is a  $k$ -dimensional  $\mathbb{F}_{q^m}$ -linear subspace of  $\mathbb{F}_{q^m}^n$  with minimum distance

$$d := \min_{\mathbf{a}, \mathbf{b} \in C, \mathbf{a} \neq \mathbf{b}} d_R(\mathbf{a}, \mathbf{b})$$

### 3. Code-based Cryptography

#### ●McNie – key generation

##### Key Generation

Generate a random  $l \times n$  generator matrix  $G'$  for a code over  $\mathbb{F}_{q^m}$

Generate the parity check matrix  $H$  of an  $[n, k]$  linear code over  $\mathbb{F}_{q^m}$  with an efficient decoding algorithm  $\Phi_H$  which can correct errors of (Hamming or rank) weight up to  $r$ .

Construct random  $n \times n$  permutation matrix  $P$  and  $(n - k) \times (n - k)$  invertible matrix  $S$  over  $\mathbb{F}_{q^m}$ .

Let  $F = G' P^{-1} H^T S$ .

- Secret key:  $(P, H, S, \Phi_H)$
- Public key:  $(G', F)$



# 3. Code-based Cryptography

## ●McNie – encryption and decryption

### Encryption

The sender generates a random vector  $\mathbf{e}$  of weight  $r$ . A message  $\mathbf{m}$  is then encrypted as  $Enc(\mathbf{m}) = (\mathbf{c}_1, \mathbf{c}_2)$  where

$$\mathbf{c}_1 = \mathbf{m}G' + \mathbf{e}$$

$$\mathbf{c}_2 = \mathbf{m}F$$

### Decryption

Let the received ciphertext be  $\mathbf{y} = (\mathbf{c}_1, \mathbf{c}_2)$ . Compute

$$c_1P^{-1}H^T - c_2S^{-1} = \mathbf{e}P^{-1}H^T.$$

Apply the decryption algorithm  $\Phi_H$  to obtain  $\mathbf{e}P^{-1}$  and multiply by  $P$  to get the error vector  $\mathbf{e}$ .

Finally,  $\mathbf{m}$  is recovered by solving the system  $\mathbf{m}G' = \mathbf{c}_1 - \mathbf{e}$ .

### 3. Code-based Cryptography

#### ●McNie2

- Gaborit proposed a message recovery attack on the McNie in 2017
- By the attack, the security level was reduced significantly
- J.-L. Kim et al. suggest the McNie2 which is upgraded by merging McNie and Dual-Ouroboros cryptosystem
- McNie2 uses another rank-metric code Gabidulin code
- It was suggested to NIST PQC Round 2, but it fails

### 3. Code-based Cryptography

#### ●McNie2 with Gabidulin – key generation

##### Key Generation

Generate a random vector  $\mathbf{u} \in \mathbb{F}_{q^m}^n$  and generate the  $l \times n$ -partial circulant matrix  $G'$  from  $\mathbf{u}$ .

Let  $H$  be a parity check matrix for a  $[2n-k, n]$  Gabidulin code  $C = Gab(\mathbf{g})$  over  $\mathbb{F}_{q^m}$  generated by  $\mathbf{g}$  such that  $H = [H_1 | H_2]$  where  $H_2$  is an  $(n-k) \times (n-k)$  invertible matrix. Let  $\Phi_H$  be an efficient decoding algorithm for  $C$  using  $H$ , which can correct errors of weight up to  $r = \lfloor \frac{n-k}{2} \rfloor$ .

Generate random  $n \times n$  permutation matrix  $P$ .

Compute  $F = G'P^{-1}H_1^T(H_2^T)^{-1}$ .

- Public Key:  $(G', F)$
- Secret Key:  $(P, H, \Phi_H)$

# 3. Code-based Cryptography

## ●McNie2 with Gabidulin – encryption and decryption

### Encryption

Generate random vectors  $\mathbf{e}_1 \in \mathbb{F}_{q^m}^n$  and  $\mathbf{e}_2 \in \mathbb{F}_{q^m}^{n-k}$  such that  $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2)$  has weight  $r$ . Compute

$$\mathbf{c}_1 = \mathbf{m}G' + \mathbf{e}_1$$

$$\mathbf{c}_2 = \mathbf{m}F + \mathbf{e}_2.$$

The message  $\mathbf{m} \in \mathbb{F}_{q^m}^l$  is encrypted as  $Enc(\mathbf{m}) = (\mathbf{c}_1, \mathbf{c}_2)$ .

### Decryption

Suppose the vector  $\mathbf{y} = (\mathbf{c}_1, \mathbf{c}_2)$  is received. Compute

$$\begin{aligned}\mathbf{c}_1 P^{-1} H_1^T - \mathbf{c}_2 H_2^T &= \mathbf{m}_1 G' P^{-1} H_1^T + \mathbf{e}_1 P^{-1} H_1^T - \mathbf{m} G' P^{-1} H_1^T (H_2^T)^{-1} H_2^T \\ &\quad - \mathbf{e}_2 H_2^T \\ &= \mathbf{e}_1 P^{-1} H_1^T - \mathbf{e}_2 H_2^T \\ &= (\mathbf{e}_1 P^{-1}, -\mathbf{e}_2) \begin{bmatrix} H_1^T \\ H_2^T \end{bmatrix} \\ &= \mathbf{e}' H^T\end{aligned}$$

Since  $\mathbf{e}' = (\mathbf{e}_1 P^{-1}, -\mathbf{e}_2)$  is of weight  $r$ , the decoding algorithm  $\Phi_H$  can be applied to obtain  $(\mathbf{e}'_1, -\mathbf{e}_2)$ .

Apply the permutation  $P$  to  $\mathbf{e}'_1 = \mathbf{e}_1 P^{-1}$  to obtain  $\mathbf{e}_1$ .

Finally, solve the system  $\mathbf{m}G' = \mathbf{c}_1 - \mathbf{e}_1$  to recover  $\mathbf{m}$ .

## 4. REDOG

## 4. REDOG

### ● REDOG (Reinforced modified Dual-Ouroboros based on Gabidulin codes)

- T.S.C. Lau et al. pointed out the selection of secret key  $S$  for the Dual-Ouroboros cryptosystem in 2021
- So we upgraded and proposed REDOG for the KpqC competition
- During 2023 to 2024, T. Lange et al. attacked REDOG several times and suggested other rank-metric decoding attacks
- Considering those tasks, we finalize the REDOG cryptosystem

### ● REDOG – key generation

Setup: Generate global parameters with integers  $m, n, \ell, r, k$  such that  $\ell < n$  and  $t_1 + \lambda t_2 \leq r \leq \left\lfloor \frac{n-k}{2} \right\rfloor$ . Output parameters =  $(m, n, \ell, k, r, \lambda, t_1, t_2)$ .

Key.Gen: Select  $H = [H_1 H_2]$ ,  $H_2 \in \text{GL}_{n-k}(\mathbb{F}_{q^m})$ , a parity check matrix of a  $[2n-k, n]$

Gabidulin code  $\mathcal{C}$ , with syndrome decoder  $\Phi$  correcting  $r$  errors where  $r = \left\lfloor \frac{n-k}{2} \right\rfloor$ . Select a full rank matrix  $M \in \mathbb{F}_{q^m}^{\ell \times n}$ . Select a  $\lambda$ -dimensional subspace  $\Lambda \subset \mathbb{F}_{q^m}$ , seen as  $\mathbb{F}_q$ -linear space, and select  $S^{-1} \in \text{GL}_{n-k}(\Lambda)$  and  $P \in \mathbb{F}_{q^m}^{n \times n}$ .

Output public key and secret key pair

$\text{pk} = (M, F = MP^{-1}H_1^T[H_2^T]^{-1}S), \text{sk} = (P, H, S, \Phi)$ .

## ● REDOG – encryption and decryption

$\text{Enc}(\text{pk}, \mathbf{m} \in \mathbb{F}_{q^m}^\ell)$ : Let  $\mathbf{m} \in \mathbb{F}_{q^m}^\ell$  be the plaintext message to be encrypted.

Generate uniformly random vector  $\mathbf{e} = (e_1, e_2) \in \mathbb{F}_{q^m}^{2n-k}$  with  $\text{wt}_R(e_1) = t_1$ , and  $\text{wt}_R(e_2) = t_2$ , where  $e_1 \in \mathbb{F}_{q^m}^n$  and  $e_2 \in \mathbb{F}_{q^m}^{n-k}$ . Compute  $\mathbf{m}' = \mathbf{m} + \mathcal{H}(\mathbf{e})$ .

Compute  $c_1 = \mathbf{m}'M + e_1$  and  $c_2 = \mathbf{m}'F + e_2$ .

Output ciphertext  $\mathbf{c} = (c_1, c_2)$ .

$\text{Dec}(\text{sk}, c = (c_1, c_2))$ : Compute

$$\begin{aligned} & c_1 P^{-1} H_1^T - c_2 S^{-1} H_2^T \\ &= \mathbf{m}' M P^{-1} H_1^T + e_1 P^{-1} H_1^T - \mathbf{m}' M P^{-1} H_1^T [H_2^T]^{-1} S S^{-1} H_2^T - e_2 S^{-1} H_2^T \\ &= e_1 P^{-1} H_1^T - e_2 S^{-1} H_2^T \\ &= (e_1 P^{-1}, -e_2 S^{-1}) \begin{bmatrix} H_1^T \\ H_2^T \end{bmatrix} \end{aligned}$$

Let  $\mathbf{e}' = (e_1 P^{-1}, -e_2 S^{-1})$ . Since  $\text{wt}_R(\mathbf{e}') \leq r$ , apply  $\Phi_H$  to obtain  $\mathbf{e}'$ .

Compute  $e_1 = e_1 P^{-1} P$  and  $e_2 = e_2 S^{-1} S$  to obtain  $\mathbf{e} = (e_1, e_2)$ .

Finally, solve the system  $\mathbf{m}'G = c_1 - e_1$  to recover  $\mathbf{m} = \mathbf{m}' - \mathcal{H}(\mathbf{e})$ .



### ● REDOG

- Added calculation by selecting non-singular matrix  $S$  in the previous version
  - In this process, several problems occurred
  - T.S.C. Lau et al. suggest to consider about the dimension of  $S$ 
    - ✓ The Lambda-dimensional subspace
    - ✓ But the subspace was not a group, so we modified the part
  - To avoid such problems, we select the matrix  $S^{-1}$  in a  $\mathbb{F}_q$ -linear space  $GL_{n-k}(\Lambda)$

# 4. REDOG

## ● REDOG

- Initially, the latest attacks on rank-metric code cryptosystem were not considered
  - We checked about BBB+, BBC+, BBB23 attacks (Bardet et al. 2023)

Security level and cost for each parameter of REDOG

Instance	$(n, k, \ell, q, m, r, \lambda, t_1, t_2)$	$AGHT$	$GRS$	$BBB+$	$BBC+$	$BBB23$	Security level
REDOG-1	(30,6,25,2,59,12,3,6,2)	198.41	226.01	140.05	144.99	145.79	128
REDOG-2	(44,8,37,2,83,18,3,12,2)	423.23	462.03	229.45	358.13	357.75	192
REDOG-3	(58,10,49,2,109,24,3,15,3)	749.01	800.30	324.24	666.09	672.91	256

## 4. REDOG

### ● REDOG

- Proposed parameters of REDOG

Instance	$(n, k, \ell, q, m, r, \lambda, t_1, t_2)$	size <sub>pk</sub>	size <sub>sk</sub>	size <sub>ct</sub>	Security level
REDOG-1	(30,6,25,2,59,12,3,6,2)	4.17KB	0.65KB	0.38KB	128
REDOG-2	(44,8,37,2,83,18,3,12,2)	13.66KB	1.43KB	0.82KB	192
REDOG-3	(58,10,49,2,109,24,3,15,3)	31.87KB	2.50KB	1.44KB	256

- This is the parameters of Classic McEliece, which is selected in NIST PQC Round 4

Variant	$n$	$m$	$t$	$k = n - mt$	pk size	sk size	Security level
mceliece6960119	6960	13	119	5413	1047KB	13.6KB	256
mceliece8192128	8192	13	128	6528	1358KB	13.75KB	256

### ●REDOG

- Use rank-metric code to efficiency – unique rank-metric code based system
- For encryption, both the generator matrix and parity check matrix are used
- Going through several attacks, REDOG became more robust
- Previously, it was mainly written in Python-based Sage language.
  - The task of implementing optimization code in C language has done

**감사합니다**