

# Summary of ELEC 433

Tom Wang

Spring, 2024

## 1 Introduction to Hamming Codes

ECC solves the problem of transmission errors. The idea is to add redundancy to the message so that the receiver can detect and correct errors.

Assume that we have a message of length  $k$ , then we encode it to a codeword  $C$  of length  $n$ . If the codeword is decrypted, then we call it  $\tilde{C}$ .

### 1.1 Things we care about

1. **Error Detection:** We want to be able to detect if there are errors in the transmission.
2. **Error Correction:** We want to be able to correct the errors in the transmission.
3. **Efficiency:** We want to be able to minimize the number of bits we need to add to the message. So minimize the overhead. ( $\frac{k}{n}$  should be as large as possible)
4. **Complexity:** We want to be able to encode and decode the message in a reasonable amount of time.
5. **Robustness:** We want to be able to correct as many errors as possible.
6. **Flexibility:** We want to be able to correct different types of errors.

## 1.2 Definitions

First of all, we need to define a code:

**Definition 1.** A code  $C$  of length  $n$  over an alphabet  $\Sigma$  is a subset of  $\Sigma^n$ . So  $C \subseteq \Sigma^n$ . The elements  $c \in C$  are called codewords.

There are two types of errors:

- **Erasures:** We know which bit is erased, but we don't know what it is. So we can **detect** and **correct** erasures.
- **Error:** We don't know which bit is wrong, but we know that there is at least one bit that is wrong. So we can **detect** errors, but we can't for sure **correct** them.

Also, we define all possible codewords to be  $C_i$  where they all belong to the same set  $\mathcal{C}$ . So  $C_i \in \mathcal{C}$  for all possible  $i$ .

## 1.3 Hamming Codes

Hamming codes are linear block codes. They are named after Richard Hamming. They are single-error correcting and double-error detecting. They are also systematic, which means that the message is part of the codeword.

Hamming code encodes a message of length 4 to a codeword of length 7. So  $k = 4$  and  $n = 7$ . Also, it encodes only binary messages, so  $\Sigma = \{0, 1\}$ .

$$(x_1, x_2, x_3, x_4) \implies (x_1, x_2, x_3, x_4, x_2 + x_3 + x_4, x_1 + x_3 + x_4, x_1 + x_2 + x_4)$$

Note that they are binary additions. So  $1 + 1 = 0$ . It is basically XOR.

Some definitions:

- **Hamming distance:** Hamming distance between  $x, y \in \Sigma^n$  is the number of positions where  $x$  and  $y$  differ. It is denoted by  $d(x, y) = \sum_{i=1}^n \mathbb{1}\{x_i \neq y_i\}$ . So adding each bit that is different.
- **Relative distance:** counts for the length of the codeword. So  $\frac{d(x, y)}{n}$ .
- **Minimum distance:** The minimum distance between any two codewords. So  $d_{min} = \min d(x, y)$ , for which  $x, y \in C, x \neq y$ . It measures the robustness of the code. The larger the minimum distance, the more errors it can correct.

- **Hamming weight:** The number of non-zero elements in a vector. So  $w(x) = \sum_{i=1}^n \mathbb{1}\{x_i \neq 0\}$ . It is the number of 1s in the vector.

## 1.4 Hamming Code Properties

**Theorem 1.** *A code with a minimum distance of  $d$  can:*

- Detect  $d - 1$  errors.
- Correct  $\lfloor \frac{d-1}{2} \rfloor$  errors.
- Detect  $d - 1$  erasures.
- Correct  $d - 1$  erasures.

## 1.5 Algorithm for Hamming Code

### 1.5.1 Error Detection

If given a codeword  $\tilde{C}$  that we assume the error is within  $d - 1$  away from one point, we can calculate the syndrome  $S$  by  $S = \tilde{C}H^T$ . If  $S = 0$ , then there is no error. If  $S \neq 0$ , then there is an error.

### 1.5.2 Error Correction

For a codeword  $\tilde{C}$ , with an error smaller than or equal to  $\lfloor \frac{d-1}{2} \rfloor$ , we can recover the original codeword by finding the closest codeword to  $\tilde{C}$ . The distance will be within  $\lfloor \frac{d-1}{2} \rfloor$ .

Note that it is not possible to have two  $C$  that are within  $\lfloor \frac{d-1}{2} \rfloor$  distance from  $\tilde{C}$ . Otherwise, the distance between the two  $C$  will be smaller than  $d$ . Here is the proof:

*Proof.*  $\lfloor \frac{d-1}{2} \rfloor \times 2 \leq d - 1 < d$ . So circle of radius  $\lfloor \frac{d-1}{2} \rfloor$  around  $\tilde{C}$  does not contain any other codewords. So there is only one codeword that is within  $\lfloor \frac{d-1}{2} \rfloor$  distance from  $\tilde{C}$ .  $\square$

To mathematically express this:

If  $\tilde{C}$  has error  $\leq \lfloor \frac{d-1}{2} \rfloor$ , then return  $C \in \mathcal{C}$  such that  $d(\tilde{C}, C) \leq \lfloor \frac{d-1}{2} \rfloor$ .

### 1.5.3 Erasure Correction

If we have a codeword  $\tilde{C}$  with  $e$  erasures, where  $e \leq d - 1$ , then we can recover the original codeword by finding the closest codeword to  $\tilde{C}$ . The distance will be within  $e$ .

## 1.6 More definitions

**Definition 2.** The *message length/dimension* of  $\mathcal{C}$  over an alphabet  $\Sigma$  is the length of the message. So  $k = \log_{|\Sigma|} |\mathcal{C}|$ .

(The log base is the size of the alphabet set, and the log is the size of the code set)

**Definition 3.** The *rate* of  $\mathcal{C}$  is the ratio of the message length to the codeword length. So  $R = \frac{k}{n}$ .

Note the rate is between 0 and 1.

- The closer it is to 1, the more efficient the code is.
- If  $R = 1$ , then the code is not redundant. So it is not an error-correcting code.
- The closer it is to 0, the more redundant the code is.
- If  $R = 0$ , then the code is not efficient. So it is not an error-correcting code.
- So we need to make trade-offs between efficiency and robustness.

We could find that  $R$  is a function:  $R(d, n, k, \Sigma)$ . So we can find the maximum rate and minimum rate for a given  $d, n, k, \Sigma$ . Basically,  $R$  has two bounds:

$$g(d, n, k, \Sigma) \leq R(d, n, k, \Sigma) \leq f(d, n, k, \Sigma)$$

## 1.7 Hamming Bound

Hamming bound is the upper bound of the rate of a code. It answers the question: How many Hamming balls can we put in  $\Sigma^n$  such that they are disjoint for a fixed  $d$ ?

The question implies that the ball radius is  $\lfloor \frac{d-1}{2} \rfloor$ . With the given information we can find that

- We will have  $|\mathcal{C}|$  disjoint balls of radius  $\lfloor \frac{d-1}{2} \rfloor$ .
- $\text{Vol}(\Sigma^n) = |\Sigma|^n$ .
- $|\mathcal{C}| \leq \frac{|\Sigma|^n}{\text{Vol}(\text{Hamming ball})}$ .

Note that we are dealing with discrete points. So we need to use the floor function which actually just counts for the points within the ball.

### 1.7.1 Proof of Hamming Bound

*Proof.* The Hamming ball in  $\Sigma^n$  of radius  $e$  about  $x \in \Sigma^n$  is the set of all points in  $\Sigma^n$  that are within  $e$  distance from  $x$ .

So  $B_{\Sigma^n}(x, e) = \{y \in \Sigma^n : d(x, y) \leq e\}$  is the set of the discrete points that are within  $e$  distance from  $x$ , including  $x$  itself.

Now we can define the volume of the ball. The volume of the ball is the number of points in the ball. So  $V_{\Sigma^n} = |B_{\Sigma^n}(x, e)|$  which is the size of the ballpoint set.

To make the calculation easier, we let  $q = |\Sigma|$ .

$$\text{Vol}_q(e, n) = 1 + \binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + \dots + \binom{n}{e}(q-1)^e = \sum_{i=0}^e \binom{n}{i}(q-1)^i$$

So since Hamming balls cannot overlap, we have:

$$\begin{aligned} |\mathcal{C}| &\leq \frac{q^n}{\text{Vol}_q(e, n)} \\ |\mathcal{C}| \times \text{Vol}_q(e, n) &\leq q^n \\ \log_q(|\mathcal{C}|) + \log_q(\text{Vol}_q(e, n)) &\leq \log_q(q^n) \\ \log_q(|\mathcal{C}|) + \log_q(\text{Vol}_q(e, n)) &\leq n \\ \log_q(|\mathcal{C}|) &\leq n - \log_q(\text{Vol}_q(e, n)) \\ \text{Rate} = \frac{\log_q(|\mathcal{C}|)}{n} &\leq 1 - \frac{\log_q(\text{Vol}_q(e, n))}{n} \end{aligned}$$

The rate here is the maximum rate. So we have the upper bound of the rate of a code. So we have the **Hamming bound**:  $\square$

### 1.7.2 Analysis of Hamming Bound for binary codes

Now that we have all the tools for binary Hamming codes:

- $q = 2$ . So  $|\Sigma| = 2$ , which is the alphabet size.
- $n = 7$ . So  $n$  is the length of the codeword.
- $k = 4$ . So  $k$  is the length of the message.
- $d = 3$ . So  $d$  is the minimum distance. Proof of this will be in the next section.

Using the above information, we find that the Hamming bound is:

$$R = 1 - \frac{\log_2(\text{Vol}_2(\lfloor \frac{3-1}{2} \rfloor, 7))}{7} = 1 - \frac{\log_2(1+7)}{7} = \frac{4}{7}$$

But we also notice that the Hamming code has a rate of  $\frac{k}{n} = \frac{4}{7}$ . So the Hamming bound is tight. So the Hamming code is the best/optimal code for this case.

## 1.8 Linear Algebra View of Hamming Codes

If we view the encoding process as a matrix multiplication it will be like this:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_2 + x_3 + x_4 \\ x_1 + x_3 + x_4 \\ x_1 + x_2 + x_4 \end{bmatrix}$$

We can write it as  $G \cdot X = C$  where  $G$  is the generator matrix. So  $G$  is a  $n \times k$  matrix.  $X$  is the message vector.  $C$  is the codeword vector.  $C = \{G \cdot X \pmod{2} : \forall x \in X, x \in \{0, 1\}^4\}$

There are three properties with it:

1.  $C$  is closed under addition. So  $\forall C_1, C_2 \in C, C_1 + C_2 \in C$ .

*Proof.*

$$C_1 + C_2 = G \cdot X_1 + G \cdot X_2 = G \cdot (X_1 + X_2)$$

Since  $X_1 + X_2 \in \{0, 1\}^4$ , so  $C_1 + C_2 \in C$ . Proved. □

2.  $\mathcal{C}$  is a linear subspace of  $\{0, 1\}^7$  of dimension 4.

*Proof.*  $\mathcal{C}$  is all the linear combinations of the columns of  $G$ . So  $\mathcal{C}$  is a linear subspace of  $\{0, 1\}^7$  of dimension 4. Proved.  $\square$

3. The distance of  $\mathcal{C}$  is the same as the minimum weight of the non-zero codewords in  $\mathcal{C}$ .

*Proof.* Recall that  $wt(x) = \sum_{i=1}^n \mathbb{1}\{x_i \neq 0\}$ . So  $wt(x)$  is the number of 1s in the vector.

$\forall C_1, C_2 \in \mathcal{C}, C_1 \neq C_2$ , then  $wt(C_1 + C_2) = wt(C_1 \oplus C_2)$ , which is just the number of different bits between  $C_1$  and  $C_2$ . So  $d(C_1, C_2) = wt(C_1 \oplus C_2)$ .

Thus if we go through all the possibilities space, we can claim that  $d_{min} = \min_{C_1, C_2 \in \mathcal{C}, C_1 \neq C_2} d(C_1, C_2) = \min_{C \in \mathcal{C}, C \neq 0} wt(C)$ . Proved.  $\square$

### 1.8.1 Parity Check Matrix

We can also define a parity check matrix  $H$  for the Hamming code. It is a  $n \times (n - k)$  matrix. So  $H \cdot C^T = 0$  where  $C$  is the codeword vector. So  $H$  is the matrix that can detect errors.

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ c_5 = x_2 + x_3 + x_4 \\ c_6 = x_1 + x_3 + x_4 \\ c_7 = x_1 + x_2 + x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Note that  $H$  is just letting the encoded codeword to XOR with its decomposition. It should be 0 if there is no error. So if there is an error, then  $H \cdot C^T \neq 0$ .

### 1.8.2 Proof of Minimum Distance

With the help of the parity check matrix, we can prove that the minimum distance of the Hamming code is 3.

*Proof.* Proof by contradiction: Suppose that  $\exists C \in \mathcal{C}$  such that  $wt(C) = 1$  or  $wt(C) = 2$ .

We can list out all the possibilities of  $C$  and find out that:  $H \cdot C^T \neq 0$ . So  $C$  is not a codeword. Contradiction. So  $\forall C \in \mathcal{C}, wt(C) \geq 3$ .

Now just need to find a codeword with weight 3. We can find that  $C = (1, 1, 1, 0, 0, 0, 0)$  is a codeword. With the property that we just proved:

*The distance of  $\mathcal{C}$  is the same as the minimum weight of the non-zero codewords in  $\mathcal{C}$ .*

So  $d_{min} = 3$ . Proved.  $\square$

## 1.9 Decoding Hamming Codes

Assume that we have a codeword  $\tilde{C}$  that is transmitted. We want to find the closest codeword to  $\tilde{C}$ .

- If  $\tilde{C}$  has no error, then we can just return  $\tilde{C}$ .
- If  $\tilde{C}$  has one error,  $\tilde{C} = C \oplus Z$  where  $Z$  is the noise such that  $wt(Z) = 1$ .

$$\begin{aligned}
 H \cdot \tilde{C}^T &= H \cdot (C \oplus Z)^T \\
 &= H \cdot C^T \oplus H \cdot Z^T && \text{Addition is commutative} \\
 &= H \cdot Z^T && \text{Since } H \cdot C^T = 0
 \end{aligned}$$

From there, we can know which bit is wrong now, and moreover, we can correct it.

$$C = \tilde{C} \oplus Z$$

XOR can be reversed by XOR again. So we can correct the error.

## 2 General Linear Block Codes

### 2.1 Finite Fields

Intuitively, a (finite) field is a (finite) set of elements that can be added, subtracted, multiplied, and divided.

Formally, we can define as follows:

**Definition 4.** A field  $\mathbb{F}$  is a set of elements, along with operations of addition and multiplication, such that  $\forall x, y, z \in \mathbb{F}$



- (Associativity)  $x + (y + z) = (x + y) + z$  and  $x \cdot (y \cdot z) = (x \cdot y) \cdot z$ .
- (Commutativity)  $x + y = y + x$  and  $x \cdot y = y \cdot x$ .
- (Distributivity)  $x \cdot (y + z) = x \cdot y + x \cdot z$ .
- (Identity):
  - $\exists '0' \in \mathbb{F}$  such that  $x + 0 = x$ .
  - $\exists '1' \in \mathbb{F}$  such that  $x \cdot 1 = x$ .
- (Inverse):
  - $\forall x \in \mathbb{F}, \exists -x \in \mathbb{F}$  such that  $x + (-x) = 0$ .
  - $\forall x \in \mathbb{F}, x \neq 0, \exists x^{-1} \in \mathbb{F}$  such that  $x \cdot x^{-1} = 1$ .

### 2.1.1 Existence of finite fields

We can examine two examples first:  $\mathbb{F}_1 = \{0, 1, 2, 3, 4\}$  with  $'+' = '+ \bmod 5'$ ,  $'\times' = '\times \bmod 5'$  and  $\mathbb{F}_2 = \{0, 1, 2, 3\}$  with  $'+' = '+ \bmod 4'$ ,  $'\times' = '\times \bmod 4'$ .

It is easy to verify associativity, commutativity, distributivity and identity for both, but let's take a closer look into the inverse.

- $\mathbb{F}_1$ :
  - 0 has no inverse for multiplication.
  - 1, 2, 3, 4 all have inverses for multiplication.
  - 0, 1, 2, 3, 4 all have inverses for addition.
- $\mathbb{F}_2$ :
  - 0 has no inverse for multiplication.
  - 1, 3 all have inverses for multiplication. But what is the inverse of 2? There is no such thing. **So  $\mathbb{F}_2$  is not a field.**
  - 0, 1, 2, 3 all have inverses for addition.

**Theorem 2.** When  $p$  is a prime number, then  $\mathbb{F}_p = \{0, 1, 2, \dots, p-1\}$  with  $'+' = '+ \bmod p'$ ,  $'\times' = '\times \bmod p'$  is a field.

## 2.2 Linear algebra's view of finite fields

Let  $\mathbb{F}$ :

- $\mathbb{F}^n = \{(x_1, x_2, \dots, x_n) : x_i \in \mathbb{F}\}$ .
- A **subspace**  $V \subseteq \mathbb{F}^n$  is a set of vectors that is closed under addition and scalar multiplication. So  $\forall u, v \in V, \lambda \in \mathbb{F}, u + v \in V, \lambda u \in V$ .
- $v_1, v_2, \dots, v_k \in \mathbb{F}^n$  are **linearly independent** if  $\lambda_1 v_1 + \lambda_2 v_2 + \dots + \lambda_k v_k = 0$  implies that  $\lambda_1 = \lambda_2 = \dots = \lambda_k = 0$ .
- For  $v_1, v_2, \dots, v_k \in \mathbb{F}^n$ , the **span** of  $v_1, v_2, \dots, v_k$  is the set of all linear combinations of  $v_1, v_2, \dots, v_k$ . So  $\text{span}(v_1, v_2, \dots, v_k) = \{\lambda_1 v_1 + \lambda_2 v_2 + \dots + \lambda_k v_k : \lambda_1, \lambda_2, \dots, \lambda_k \in \mathbb{F}\} = \{\sum_i \lambda_i v_i : \lambda \in \mathbb{F}\}$ .
- A **basis** for a subspace  $V \subseteq \mathbb{F}^n$  is a set of linearly independent vectors that span  $V$ .
- The **dimension** of a subspace  $V \subseteq \mathbb{F}^n$  is the number of vectors in a basis for  $V$ .

## 2.3 Linear Block Codes over Finite Fields

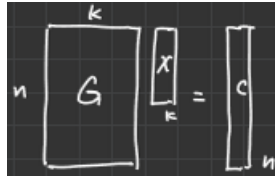
**Definition 5.** A linear block code  $\mathcal{C}$  of length  $n$  and dimension  $k$  over a finite field  $\mathbb{F}$  is a  $k$ -dimension linear subspace of  $\mathbb{F}^n$ . So  $\mathcal{C} \subseteq \mathbb{F}^n$ . The elements  $c \in \mathcal{C}$  are called codewords.

Note that if we connect to previous definitions, it is the same as block length  $n$  and message length  $k$  over alphabet  $\Sigma = \mathbb{F}$ .

$\mathcal{C}$  is a “linear code”  $\implies \mathcal{C}$  is a linear subspace of  $\mathbb{F}^n$

### 2.3.1 Generator Matrix

**Definition 6.** Let  $\mathcal{C} \subseteq \mathbb{F}^n$  be a linear block code of length  $n$  and dimension  $k$ . A generator matrix  $G$  for  $\mathcal{C}$  is a  $n \times k$  matrix such that  $\mathcal{C} = \{G \cdot X : X \in \mathbb{F}^k\}$ .



The diagram shows a large rectangle labeled 'G' with 'n' on the left and 'k' on top. To its right is a smaller vertical rectangle labeled 'X' with 'k' on the left. An equals sign follows, then another vertical rectangle labeled 'c' with 'n' on the right.

$$[G] = \begin{bmatrix} I_k \\ A \end{bmatrix}$$

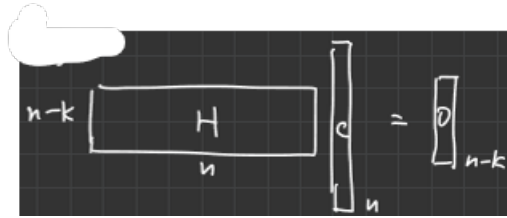
Observations:

- Any linear code has a generator matrix. (Choose any basis for the code and put the basis vectors as the columns of the generator matrix.)
- The generator matrix is not unique. (Different bases will give different generator matrices.)

A generator like the above is called a **systematic generator matrix** where  $I_k$  is the identity matrix of size  $k$  and  $A$  is a  $(n - k) \times k$  matrix. It is called systematic because the message is part of the codeword.

### 2.3.2 Parity Check Matrix

**Definition 7.** Let  $\mathcal{C} \subseteq \mathbb{F}^n$  be a linear block code of length  $n$  and dimension  $k$ . A parity check matrix  $H \in \mathbb{F}^{(n-k) \times k}$  for  $\mathcal{C}$  is a  $(n - k) \times n$  matrix such that  $\mathcal{C} = \{C \in \mathbb{F}^n : H \cdot C = 0\}$ .  $H$  is the kernel matrix of  $G$ .



$$[H] = [B_{(n-k) \times k} \quad I_k]$$

Observations:

- Any linear code has a parity check matrix.
- The parity check matrix is not unique.

A parity check matrix like the above is called a **systematic parity check matrix** where  $I_{(n-k)}$  is the identity matrix of size  $(n - k)$  and  $B$  is a  $(n - k) \times k$  matrix. It is called systematic because the message is part of the codeword.

Note that the minimum distance of a code is equal to the number of linearly dependent columns in the parity matrix:

*Proof.* Since we know  $H \cdot \vec{c} = 0, \forall \vec{c} \in \mathcal{C}$ , we start by assuming the minimum weight of a non-zero code  $c^*$  is  $d$ . Since it is non-zero, the columns with 1 in  $c^*$  must be linearly dependent to get the sum to be zero. Thus, the minimum number of linearly dependent columns is  $d$ . Thus, the minimum distance of the code is  $d$ .  $\square$

### 2.3.3 Relationship between G and H

Question:

Let  $\mathcal{C}$  be a linear code of dimension  $k$  and length  $n$  over  $\mathbb{F} = \{0, 1\}$ . Let  $G$  be a generator matrix for  $\mathcal{C}$  and  $H$  be a parity check matrix for  $\mathcal{C}$ . What is the relationship between  $G$  and  $H$ , or more specifically what is the relationship between  $A$  and  $B$ ?

Let the original message be  $X$ . Note that  $A \cdot X$  is the parity bits.

Then we can write the codeword as  $C = G \cdot X = \begin{bmatrix} I_k \\ A \end{bmatrix} \cdot X = \begin{bmatrix} X \\ (A \cdot X) \end{bmatrix}$ .

Also, we know that  $H \cdot C = 0$ . So

$$\begin{bmatrix} B_{(n-k) \times k} & I_k \end{bmatrix} \cdot \begin{bmatrix} X_{k \times k} \\ (A \cdot X)_{(n-k) \times k} \end{bmatrix} = 0 \implies B \cdot X + I \cdot (A \cdot X) = 0$$

So  $B + A = 0$  note that the addition is the addition defined in the field  $\mathbb{F} = \{0, 1\}$ . So  $B + A = 0 \implies B = A$  in a binary sense.

### 2.3.4 useful facts about linear code:

Let  $\mathcal{C}$  be a linear code of dimension  $k$  and length  $n$  over  $\mathbb{F} = \{0, 1\}^n$ . Let  $G$  be a generator matrix for  $\mathcal{C}$  and  $H$  be a parity check matrix for  $\mathcal{C}$ .

1.  $H \cdot G = 0, \forall C \in \mathcal{C}, C = G \cdot X, H \cdot C = 0$ . This is because  $H \cdot C = H \cdot (G \cdot X) = (H \cdot G) \cdot X = 0 \cdot X = 0$ .
2. For  $\mathbb{F}_2$ , any linear code has a systematic generator matrix  $\begin{bmatrix} I_k \\ A \end{bmatrix}$  and systematic parity check matrix  $\begin{bmatrix} A & I_{n-k} \end{bmatrix}$ .
3. Any column of  $G$  is a codeword. This is because  $H \cdot G = 0$  which implies each column of  $G$  has  $H$  as a kernel. So each column of  $G$  is a codeword.
4. The distance of  $\mathcal{C}$  is the same as the minimum weight of the non-zero codewords in  $\mathcal{C}$ . This is because  $\min d(C_1, C_2) = \min d(C_1 - C_2, 0) = \min wt(C_1 \oplus C_2) = \min wt(C_1 - C_2)$ . Now if one of  $C_1$  and  $C_2$  is 0, then  $C_1 - C_2 = C_1$  (if  $C_2$  is zero here)
5. let  $\mathcal{C}$  have distance  $d$ .
  - there are some  $d$  columns of  $H$  that are linearly dependent.

- Any  $\leq d - 1$  columns of  $H$  are linearly independent.

*Proof.* By fact 4,  $\exists$  some  $C \in \mathcal{C}$  with  $wt(C) = d, HC = 0$ . Thus these  $d$  columns of  $H$  are linearly dependent to add up to 0 somehow.

$\forall y$  with  $wt(y) \leq d - 1, Hy \neq 0$  So  $Hy$  is a linearly independent set of columns of  $H$ .  $\square$

## 2.4 Decoding Linear Block Codes

### 2.4.1 Error Correction

Let  $\mathcal{C}$  be a linear code of distance  $d$ . Then  $\mathcal{C}$  can correct  $\lfloor \frac{d-1}{2} \rfloor$  errors.

Input:  $\tilde{C}$ , a codeword

1. If  $\tilde{C} \in \mathcal{C}$ , then return  $\tilde{C}$ . (No errors or too many errors)
2. else Find  $C \in \mathcal{C}$  such that  $d(\tilde{C}, C) \leq \lfloor \frac{d-1}{2} \rfloor = e$ . If such  $C$  exists, then return  $C$ . The Algorithm is as follows:
  - (a) for  $i = 1, 2, \dots, e$
  - (b) for  $S \subseteq \{1, 2, \dots, n\}$  with  $|S| = i$  ( $|S|$  is the number of non-zero elements in the error vector.  $S$  is the position of non-zero elements)
  - (c) let  $e_{n \times 1}$  be the error vector with all possible value in  $\mathbb{F}$  code space. in the  $i$ th position and 0 elsewhere. (Test all combinations of possible values in the code space with positions specified by  $S$ )
  - (d) if  $\tilde{C} - e \in \mathcal{C}$ , then return  $\tilde{C} - e$ .
  - (e) If all possibilities are explored, then return “fail”.

### 2.4.2 Error Detection

Let  $\mathcal{C}$  be a linear code of parity matrix  $H$ .

Input:  $\tilde{C}$ . A codeword

1. If  $H \cdot \tilde{C} = 0$ , then return “no error”.
2. else return “error”.

Remark: This algorithm can guarantee to detection of  $d - 1$  errors. For more than  $d$  errors, it might or might not detect them. It depends on the error pattern. When we detect them, it might be because we thought the correct code is another one that is within the Hamming ball of radius  $0 \leq e \leq d - 1$ . So we can detect it.

Note: Any error pattern corresponding to linearly independent columns of  $H$  can be detected. For example,  $\text{ENC}(x_1, x_2, x_3) = (x_1, x_2, x_3, x_1 + x_2 + x_3)$ ,

$H = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$  Can detect any odd number of errors. Because any odd number of errors will correspond to linearly independent columns of  $H$ .

## 2.5 Generalize Hamming Codes

Generalized Hamming codes are linear block codes.  $(2^r - 1, 2^r - r - 1, 3)_2$ -code. So  $n = 2^r - 1$ ,  $k = 2^r - r - 1$ ,  $d = 3$ .

The  $(7, 4, 3)$  version of Hamming code we see before is  $r = 3$ .

$H_3 = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$  Note that each column is just a binary representation of number  $[1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7]$

For integer  $r \geq 3$ , define  $H_r$  as a  $r \times (2^r - 1)$  matrix with columns as the binary representation of the numbers  $1, 2, \dots, 2^r - 1$ .

Then  $\mathcal{C}_r = \{C \in \{0, 1\}^{2^r - 1} : H_r \cdot C^T = 0\}$  is a linear block code of length  $2^r - 1$  and dimension  $2^r - r - 1$ . It is called the  $(2^r - 1, 2^r - r - 1, 3)_2$ -code.

Remark: Hamming bound of  $(2^r - 1, 2^r - r - 1, 3)_2$ -code is

$$1 - \frac{r}{2^r - 1} = \frac{2^r - r - 1}{2^r - 1} = \frac{k}{n} \leq 1 - \frac{\log_q(\text{Vol}_q(\lfloor \frac{d-1}{2} \rfloor, n))}{n} = 1 - \frac{\log_2(2^r)}{2^r - 1} = 1 - \frac{r}{2^r - 1}$$

Achieved by equality, the so-called perfect code.

## 2.6 Dual of a linear code

Dual space in an Euclidean space is the space of all vectors that are orthogonal to a given subspace. For example, if  $S$  is  $z$ -axis, then the dual space of  $S$  is the  $xy$ -plane.  $S^\perp = \{v \in \mathbb{R}^3 : v \cdot s = 0, \forall s \in S\}$ .

$\mathcal{C}$  is a linear code  $\iff \mathcal{C}$  is a linear subspace of  $\mathbb{F}_q^n$ . Let  $\mathcal{C} = \{c : Hc = 0\}$

Then the dual code  $\mathcal{C}^\perp$  is defined as the dual subspace in  $\mathbb{F}_q^n$ . Or equivalently,  $\mathcal{C}^\perp$  has generator matrix  $H^T$ .

So  $\mathcal{C}^\perp = \{v \in \mathbb{F}^n : v \cdot c = 0, \forall c \in \mathcal{C}\}$  is the dual of  $\mathcal{C}$ .

### 2.6.1 Dual of Hamming Code - Simplex code

Parity check matrix for  $(2^r - 1, 2^r - r - 1, 3)_2$ -code is

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

when  $r = 3$ . Note that the columns increase by 1 in binary. The parity matrix is not unique. It is different from the one with the identity matrix and  $A$  matrix. Note that as long as they span the same space, they are the same.

The generator matrix  $G$  for simplex code  $(2^r - 1, r, 2^{r-1})$  is

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

So using the  $n, k, d$  from the original code, simplex code should be  $(n, n - k, 2^{r-1})$  need to prove the last one later.

Since every column of  $G$  is a codeword, the simplex codeword space is a linear combination of the  $G$  columns.

$$\mathcal{C} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Not the fact that all non-zero codewords have weight  $2^{r-1}$ .

### 2.6.2 Hadamard Code

The generator matrix is  $G = [\vec{0} \ H_r]^T$ . So add a 0 column to the front of the parity matrix.

Then  $G$  has the top row to be zero which means the Hadamard code is  $(2^r, 2^r - r, 2^{r-1})$ -code. So the simplex code is a subcode of the Hadamard code. Hadamard codeword space is just one more 0 at the first row of the simplex codeword space. Note that it is all zero, the weight is the same.

The only difference is that the weight is exactly the half of length. So it introduces **symmetry**.

Proposition:  $\mathcal{C}_{sim,r}$  and  $\mathcal{C}_{had,r}$  both have distance  $2^{r-1}$ .

## 3 Polynomial Over finite fields

Idea: When we have a 2nd-degree polynomial, we can find it using 3 points. So if we use this polynomial to encode a message to get 4 points, then we can correct 1 error.

**Definition 8.** A univariate **polynomial** in variable  $x$  over a field  $\mathbb{F}_q$  of degree  $d$  is of the form:

$$f(X) = a_0 + a_1x + a_2x^2 + \cdots + a_dx^d$$

where  $a \in \mathbb{F}_q$  and  $a_d \neq 0$ .

The collection of all polynomials over  $\mathbb{F}_q$  of degree  $\leq d$  is denoted by  $\mathbb{F}_q[X]_{\leq d}$ .

For example, over  $\mathbb{F}_3$ ,  $f(x) = x^2 - 1$ ,  $g(x) = x^3 + 2x^2 + 2x$

- Addition:  $f(x) + g(x) = x^3 + x^2 + 2x - 1$
- Multiplication:  $f(x) \cdot g(x) = x^5 + x^4 + x^3 + x^2 - x$
- Division:  $f(x) \div g(x) = (x + 2)f(x) + 2$  (need to use long division) then we get a quotient polynomial  $(x + 2)$  and a remainder polynomial  $(2)$ .

### 3.1 Roots of Polynomials

Facts: A nonzero polynomial of degree  $\leq d$  has at most  $d$  roots in  $\mathbb{F}_q$ .

$$|\{x \in \mathbb{F}_q : f(x) = 0\}| \leq d$$

For example in  $\mathbb{F}_3$ :



- $f(x) = x^2 - 1$  has 2 roots: 1 and 2.
- $f(x) = x^2 + 2x + 1 = (x + 1)^2$  has 1 root: 2. Not that in  $\mathbb{F}_3$ ,  $-1 = 2$ , since  $1 + 2 = 1 - 1 = 0 \pmod{3}$ . So  $f(2) = 0$
- $f(x) = x^2 + 1$  has no root. Note that the same polynomial in different fields has different behavior. In  $\mathbb{F}_5$ ,  $f(x) = x^2 + 1$  has 2 roots: 2 and 3.

Now we can prove the fact:

*Proof.* If  $f(x)$  which has a degree of  $d$  has  $d + 1$  roots  $\alpha_1, \alpha_2, \dots, \alpha_{d+1}$ , then  $f(x) = (x - \alpha_1)(x - \alpha_2) \dots (x - \alpha_{d+1})$ . But this is a polynomial of degree  $d + 1$ . Contradiction.  $\square$

### 3.2 Vandelnoode Matrices

**Definition 9.** A Vandelnoode matrix is of the form:

$$V = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \cdots & \alpha_1^m \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \alpha_{n-1} & \alpha_{n-1}^2 & \cdots & \alpha_{n-1}^m \end{bmatrix}_{n \times (m+1)}$$

for distinct  $\alpha_1, \alpha_2, \dots, \alpha_{n-1} \in \mathbb{F}_q$ .  $v_{ij} = \alpha_i^j$ .

Fact: The square Vandelnoode matrix is invertible.  $V \cdot V^T = I$ .

Before starting, need to show a linear algebra fact:

$(V\vec{a} \implies \vec{a} = \vec{0})$  implies that  $V$  is invertible.

*Proof.* Suppose that  $a$  is a non-zero vector such that  $V \cdot a = 0$ . Then  $\exists a^{-1} \in \mathbb{F}_q$  such that  $a \cdot a^{-1} = 1$ . So  $0 = V \cdot a \cdot a^{-1} = V \cdot 1 = V$ . Then  $V$  must be zero.

Thus it is only possible that  $V \cdot a = 0$  if  $a = 0$ . So  $V$  is invertible.  $\square$

Now we can prove the fact:

*Proof.* Let  $m = n - 1$  so that it is a square matrix. We want to show that  $V \cot \vec{a} \implies \vec{a} = \vec{0}$

$$V \cdot \vec{a} = \begin{bmatrix} \sum_{i=0}^{n-1} a_i \alpha_1^i \\ \sum_{i=0}^{n-1} a_i \alpha_2^i \\ \vdots \\ \sum_{i=0}^{n-1} a_i \alpha_n^i \end{bmatrix} = \begin{bmatrix} f(\alpha_1) \\ f(\alpha_2) \\ \vdots \\ f(\alpha_n) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Note that  $f(x) = a_0 + a_1x + a_2x^2 + \dots + a_{n-1}x^{n-1}$  is a polynomial of degree  $n - 1$  with  $n$  roots. If  $f(x) \neq 0$ , then  $f(x)$  has at most  $n - 1$  roots which means  $V \cdot a \neq 0$ . Contradiction. So  $f(x) = 0$ . So  $a_0 = a_1 = \dots = a_{n-1} = 0$ . So  $\vec{a} = \vec{0}$ . Thus Vandemnoode matrix is invertible.  $\square$

### 3.3 Polynomial Interpolation over $\mathbb{F}_q$

**Theorem 3.** Given  $(\alpha_i, y_i) \in \mathbb{F}_q \times \mathbb{F}_q$ , for  $i = 1, 2, \dots, d$  with distinct  $\alpha_i$ , there is a unique polynomial  $f(x) \in \mathbb{F}_q[x]$  such that  $f(\alpha_i) = y_i$ , for  $i = 1, 2, \dots, d$ .

*Proof.*  $f(x) = a_0 + a_1x + a_2x^2 + \dots + a_dx^d$

$$V \cdot \vec{a} = \vec{y} \implies \vec{a} = V^{-1} \cdot \vec{y}, \vec{a} \in \mathbb{F}_q$$

$V$  is invertible and  $y$  is the given points.  $\square$

Fact:  $\forall$  functions  $f : \mathbb{F}_q \rightarrow \mathbb{F}_q$ ,  $f$  is given by degree  $q - 1$  polynomial.

Remark: In  $\mathbb{R}$ ,  $f : \mathbb{R} \rightarrow \mathbb{R}$ ,  $\exists$  non-polynomial functions. However, in **finite**  $\mathbb{F}_q$ ,  $\forall$  functions  $f : \mathbb{F}_q \rightarrow \mathbb{F}_q$  is given by a polynomial of degree  $\leq q - 1$ . It has something to do with the fact that it contains only finite elements.

*Proof.* Consider  $\alpha_i \in \mathbb{F}_q$  for  $i = 1, 2, \dots, q - 1$ . Given  $(\alpha_i, f(\alpha_i))$  there is a unique polynomial  $f(x) \in \mathbb{F}_q[x]$  of degree  $\leq q - 1$  which interpolates these points.  $\square$

### 3.4 Reed-Solomon Codes

Basic idea: codewords are of the form  $(f(\alpha_1), f(\alpha_2), \dots, f(\alpha_n))$  for some polynomial  $f(x)$  of degree  $\leq n - 1$ .

Note that if we choose  $f$  to be a low-degree polynomial of degree  $k - 1$ , then we can recover the message from  $k$  points. So we can correct  $n - k$  errors.

#### 3.4.1 Definition

**Definition 10.** Let  $q \geq n \geq k$  be integers. Let  $\vec{\alpha} = \alpha_1, \alpha_2, \dots, \alpha_n \in \mathbb{F}_q$  be distinct. The Reed-Solomon code  $\mathcal{C}_{RS}(\vec{\alpha}, n, k)$  over  $\mathbb{F}_q$  with evaluation points  $\vec{\alpha}$  is defined as follows:

$$\mathcal{C}_{RS}(n, k) = \{(f(\alpha_1), f(\alpha_2), \dots, f(\alpha_n)) : f(x) \in \mathbb{F}_q[x], \deg(f(x)) \leq k - 1\}$$

For example:  $q = 3, n = 3, k = 2, \vec{\alpha} = (0, 1, 2)$ . Then  $\mathcal{C}_{RS}(\vec{\alpha}, 3, 2) = \{(f(0), f(1), f(2)) : f(x) \in \mathbb{F}_3[x], \deg(f(x)) \leq 1\}$ . List out valid polynomials  $\{a_0 + a_1x : a_0, a_1 \in \mathbb{F}_3\} = \{0, 1, 2, x, x+1, x+2, 2x, 2x+1, 2x+2\}$

If we list out possible messages for input  $(0, 1, 2)$

Encoding polynomial  $\rightarrow$  Codeword

$$0 \rightarrow (0, 0, 0)$$

$$x \rightarrow (0, 1, 2)$$

$$2x + 1 \rightarrow (1, 0, 2)$$

Note that the Reed-Solomon code is still linear. It is a linear subspace of  $\mathbb{F}_q^n$ .

$$C_1 + C_2 = (f_1(\alpha_1) + f_2(\alpha_1), \dots, f_1(\alpha_n) + f_2(\alpha_n)) = (f_3(\alpha_1), f_3(\alpha_2), \dots, f_3(\alpha_n))$$

### 3.4.2 Generator Matrix

Generator matrix of  $\mathcal{C}_{RS}(\vec{\alpha}, n, k)$  is a  $k \times n$  matrix  $G$  such that  $\mathcal{C}_{RS}(\vec{\alpha}, n, k) = \{G \cdot X : X \in \mathbb{F}_q^k\}$ .

We can achieve this by using the Vandemnoode matrix.

$$G_{RS} = V = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \cdots & \alpha_1^{k-1} \\ 1 & \alpha_2 & \alpha_2^2 & \cdots & \alpha_2^{k-1} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \alpha_n & \alpha_n^2 & \cdots & \alpha_n^{k-1} \end{bmatrix}_{k \times n}$$

Same idea:  $f(\vec{\alpha}) = V \cdot \vec{a}$  for some  $\vec{a} \in \mathbb{F}_q^k$ .

### 3.4.3 Distance of RS code

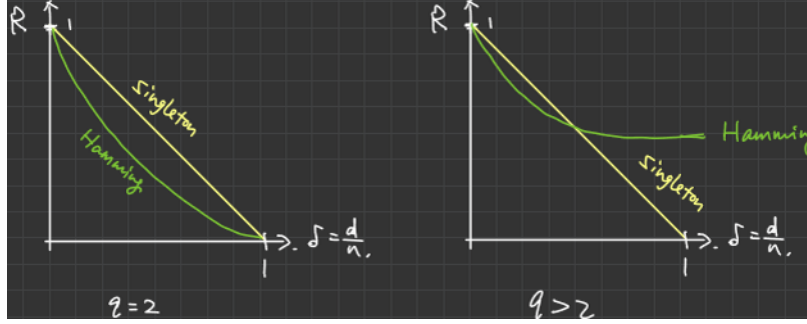
The distance of  $\mathcal{C}_{RS}(\vec{\alpha}, n, k)$  is  $d = n - k + 1$ .

*Proof.* RS codes are linear. It suffices to show that the minimum weight of non-zero is  $d = n - k + 1 \iff$  the maximum number of zeros of  $c \in \mathcal{C}, c \neq 0$  is  $n - d = k - 1$ .  $\iff$  Any polynomial has at most  $k - 1$  roots.  $\iff$  Any polynomial of degree  $\leq k - 1$  has at most  $k - 1$  roots.

We know the last statement is true □

### 3.5 The Singleton bound

**Theorem 4.** If  $\mathcal{C}$  is an  $(n, k, d)_q$ -code, then  $d \leq n - k + 1$  or  $k \leq n - d + 1$ .



Hamming bound is tighter than Singleton bound when  $q = 2$ . But when  $q > 2$ , Singleton bound is tighter than Hamming bound. Singleton bound is not dependent on  $q$  however, Hamming bound is dependent on  $q$ . So Singleton bound is a better fit for Reed-Solomon codes.

*Proof.* Let  $c \in \mathcal{C}$ ,  $c = (c_1, c_2, \dots, c_n)$ . Then we throw away the last  $d - 1$  bits and let  $\phi(c) = (c_1, c_2, \dots, c_{n-d+1})$ . Then consider:  $\mathcal{C}' = \{\phi(c) : c \in \mathcal{C}\}$ ,  $\mathcal{C}' \subseteq \mathbb{F}_q^{n-d+1}$ .

Claim 1:  $|\mathcal{C}| = |\mathcal{C}'|$  We can show it by contradiction:

If the claim is false, then  $\exists c_1, c_2 \in \mathcal{C}$  such that  $\phi(c_1) = \phi(c_2)$ . Then  $c_1$  and  $c_2$  differ in at most  $d - 1$  positions. So  $d(c_1, c_2) \leq d - 1$ . Contradiction.

There must be at least a one-bit difference among all  $c \in \mathcal{C}$ . So the two sets have the same cardinality.

Claim 2:  $|\mathcal{C}'| = q^{n-d+1}$

This is because  $\mathcal{C}$  is a linear subspace of  $\mathbb{F}_q^n$ , taking out the last  $d - 1$  bits is still a linear subspace of  $\mathbb{F}_q^{n-d+1}$ .

Thus  $|\mathcal{C}| = |\mathcal{C}'| = q^{n-d+1}$ . So  $|\mathcal{C}| \leq q^{n-d+1}$ . So  $d \leq n - k + 1$  or  $k \leq n - d + 1$ .

We can claim that singleton bound is tight for RS code.  $\square$

So RS is optimal in the sense that it achieves the Singleton bound by equality.

#### 3.5.1 MDS code

**Definition 11.** A linear  $(n, k, d)_q$ -code is called a maximum distance separable (MDS) code if  $d = n - k + 1$ .

So RS code is MDS.

Property: MDS  $\iff$  every  $k \times k$  submatrix of the generator matrix  $G$  is invertible (full rank).

So you can pick any row of  $G$  and they are all independent of each other.

*Proof.*  $d = n - k + 1 \iff$  can fill in any  $n - k$  erasures.  $\iff$  any  $k$  remaining rows of  $G$  forms a full rank matrix.  $\iff$  every  $k \times k$  submatrix of  $G$  is invertible.  $\square$

Property: Let  $\mathcal{C}$  be MDS  $(n, k, d)_q$ -code. Then any  $k$  position of  $c \in \mathcal{C}$  determines  $c$  uniquely. So we can recover the message from  $k$  points.

It is basically a restatement of the above property

### 3.6 Multiplicative group

**Definition 12.** Let  $(\mathbb{F}_q, +, \times)$  be a finite field with addition and multiplication. Let  $\mathbb{F}_q^*$  be the non-zero elements of  $\mathbb{F}_q$  with only multiplication operations. So  $(\mathbb{F}_q^*, \times)$  is a **multiplicative group**.

- $\forall a, b \in \mathbb{F}_q^*, a \times b \in \mathbb{F}_q^*$
- $\forall a \in \mathbb{F}_q^*, \exists a^{-1} \in \mathbb{F}_q^*$  such that  $a \times a^{-1} = 1$

For example:  $\mathbb{F}_5 = \{0, 1, 2, 3, 4\}$ ,  $\mathbb{F}_5^* = \{1, 2, 3, 4\}$

Fact:  $\mathbb{F}_q^*$  is a **cyclic group**. In cyclic group,  $\exists \gamma \in \mathbb{F}_q^*$  such that  $\mathbb{F}_q^* = \{\gamma^0, \gamma^1, \gamma^2, \dots, \gamma^{q-2}\}$   
Call  $\gamma$  as primitive element of  $\mathbb{F}_q^*$ . Note that  $\gamma^{q-1} = 1 = \gamma^0$  because it is a group  
 $\implies \gamma^{q-1} \times \gamma = \gamma^0 \times \gamma = \gamma$ .

For example:  $\mathbb{F}_5^* = \{1, 2, 3, 4\} = \{2^0, 2^1, 2^2, 2^3\}$  But 4 is not primitive. It is missing 2 and 3. 2 is primitive. It has all elements of  $\mathbb{F}_5^*$ .

Conjecture: For any prime  $p$ , there exists a primitive element in  $\mathbb{F}_p^*$ . Beware of the definition of a finite field,  $q$  needs to be a prime for  $\mathbb{F}_q$  to be a valid finite field.

Fact:  $\forall 0 < d < q - 1, \sum_{\alpha \in \mathbb{F}_q^*} \alpha^d = 0$ .

*Proof.* Let  $\gamma$  be a primitive element of  $\mathbb{F}_q$ . Solve using the sum of geometric series.

$$\sum_{\alpha \in \mathbb{F}_q^*} \alpha^d = \sum_{\alpha \in \mathbb{F}_q^*} \alpha^d = \sum_{j=0}^{q-2} (\gamma^j)^d = \frac{\gamma^d(1 - (\gamma^d)^{q-1})}{1 - \gamma^d} = 0$$

$r^d \neq 1$  for  $0 < r < q - 1$ .  $\square$

### 3.7 Dual view of RS code

**Proposition:** Let  $n = q - 1$  and let  $\gamma$  be a primitive element of  $\mathbb{F}_q$ .

$RS_q((\gamma^0, \gamma^1, \dots, \gamma^{q-2}), n, k) = \{(c_0, c_1, \dots, c_{n-1}) : C(\gamma^j) = 0\}$  where  $C(x) = c_0 + c_1x + \dots + c_{n-1}x^{n-1}$

*Proof.* Let  $f(x) = \sum_{i=0}^{k-1} a_i X^i$ . An RS codeword is  $f(\gamma^0), f(\gamma^1), \dots, f(\gamma^{n-1}) = (c_0, c_1, \dots, c_{n-1})$ .

Then  $C(\gamma^j) = \sum_{l=0}^{n-1} c_l \gamma^{jl} = \sum_{l=0}^{n-1} f(\gamma^l) \gamma^{jl} = f(\gamma^j) \sum_{l=0}^{n-1} \gamma^{jl} = \sum_{l=0}^{n-1} (\sum_{i=0}^{k-1} a_i \gamma^{jl}) = \sum_{i=0}^{k-1} a_i \sum_{l=0}^{n-1} (\gamma^l)^{i+j} = 0$ .

Show subset here for some reason, revisit later!!!!!!!!!!!!

□

### 3.8 Parity check matrix of RS code

**Corollary:** The parity check matrix for  $RS_q((\gamma^0, \gamma^1, \dots, \gamma^{n-1}), n, k)$  is

$$H = \begin{bmatrix} 1 & \gamma & \gamma^2 & \dots & \gamma^{n-1} \\ 1 & \gamma^2 & \gamma^4 & \dots & \gamma^{2(n-1)} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & \gamma^{n-k} & \gamma^{2(n-k)} & \dots & \gamma^{(n-k)(n-1)} \end{bmatrix}$$

*Proof.* The parity check matrix is the generator matrix of the dual code. So we can use the generator matrix of the original code to find the parity check matrix of the dual code.

$$H\vec{C} = \vec{0} = \sum_{i=0}^{n-1} c_i \gamma^{ji} = \vec{C}(\gamma^j) = 0$$

Not done yet!!!!!!!!!!!!

□

### 3.9 Generalized Reed-Solomon Codes

**Definition 13.** A generalized RS code  $GRS_q(\vec{\alpha}, n, k, \vec{\lambda})$  where  $\forall \lambda_i \in \vec{\lambda} = (\mathbb{F}_q^*)^n$  is defined as follows:

$$GRS_q(\vec{\alpha}, n, k, \vec{\lambda}) = \{(\lambda_0 f(\alpha_1), \dots, \lambda_{n-1} f(\alpha_n)) : f(x) \in \mathbb{F}_q[x], \deg(f(x)) \leq k-1\}$$

where  $\gamma_i = (\lambda_i \prod_{j \neq i} (\alpha_i - \alpha_j))^{-1}$  is a primitive element of  $\mathbb{F}_q^*$ .

**Theorem 5.** For any  $\vec{\alpha}$  and  $\vec{\lambda}$ ,  $\exists \vec{\gamma} \in \mathbb{F}_q^*$  such that

$$GRS_q(\vec{\alpha}, n, k, \vec{\lambda})^\perp = GRS_q(\vec{\alpha}, n, n-k, \vec{\gamma})$$

### 3.10 Decoding of RS code

Question: Consider  $RS_q(\vec{\alpha}, n, k)$  and assume the number of error  $e \leq \lfloor \frac{n-k}{2} \rfloor = \lfloor \frac{d-1}{2} \rfloor \iff d = n + 1 - k$ .

There are lots of possibilities for the error. How to decode the message? How do we decide which polynomial is the correct one? How to do it efficiently?

#### 3.10.1 Berlekamp-Welch Algorithm

Idea: Consider an error locator polynomial  $L(x) = \prod_{i: \tilde{c}_i \neq f(\alpha_i)} (x - \alpha_i)$ .

Observe that  $\forall i, \tilde{c}_i = f(\alpha_i)L(\alpha_i)$ . So  $L(x)$  is a polynomial of degree  $\leq e$ .

- For index  $i$  where error happened, both sides equal zero.
- For index  $i$  where no error happened,  $\tilde{c}_i = f(\alpha_i)$ .

Algorithm:

1. Find a monic (leading coefficient is 1) polynomial  $L(x)$  and a polynomial  $Q(x)$  with degree  $\leq e + k - 1$  such that  $\tilde{c}_i L(\alpha_i) = Q(\alpha_i)$  for all  $i$ .
2. Let  $f(x) = \frac{Q(x)}{L(x)}$ . Then  $f(x)$  is the correct polynomial if  $\Delta(C(f), \tilde{c}) \leq e$ . Otherwise, fail

Refer to her notes for details.

### 3.11 Pros and Cons of RS code

Pros:

- achieves Singleton bound  $k \leq n - d + 1$ . RS code meets equality of formula.
- has an efficient decoding algorithm

Cons:

- RS code requires  $q \geq n \geq k$  and  $q$  is a prime. So it is not suitable for all parameters. For example, RS code exists for large alphabet size  $q$ .

## 4 Existence of finite fields

**Theorem 6.** *For any prime power  $p^m$ , there exists a finite field  $\mathbb{F}_{p^m}$  with  $p^m$  elements. There are no other finite fields.*

Remark: For  $\mathbb{F}_{p^m}$ , it has elements from  $\mathbb{F}_p[X]$ . Addition and multiplication are defined by modulo  $p$  and modulo  $f(x)$  respectively. Multiplication is related to “irreducible polynomial”.

For example:  $f(x), g(x) \in \mathbb{F}_{p^m}$

$$f(x) \cdot g(x) = q(x)E(x) + r(x)$$

Note the similarity to the Euclidean division.  $q(x)$  is the quotient and  $r(x)$  is the remainder.  $r(x)$  has degree less than  $f(x)$ . Then  $E(x)$  is the irreducible polynomial.

### 4.1 Irreducible Polynomial

An irreducible polynomial is like a prime number in the polynomial field. It is the building block of the finite field.

**Definition 14.** *A polynomial  $f(x) \in \mathbb{F}_p[x]$  is called **irreducible** if for any  $g_1(x), g_2(x) \in \mathbb{F}_p[x]$ ,  $f(x) = g_1(x)g_2(x)$ , it implies that  $\min\{\deg(g_1(X)), \deg(g_2(x))\} = 0$*

It is basically a polynomial that cannot be factored into two polynomials of lower degree. It can only be factored into a scalar and a polynomial of the same degree.

For example:

- $2(x + 1)$  is irreducible in  $\mathbb{F}_3[x]$
- Over  $\mathbb{F}_2[x]$ ,  $x^2 + 1 = (x + 1)^2$  not irreducible (has a root which is 1)
- Over  $\mathbb{F}_2[x]$ ,  $x^2 + x + 1$  is irreducible (no root)

Is it enough to check all possible  $\alpha \in \mathbb{F}_q$  to see if  $f(\alpha) = 0$ ? No, because it is not enough to check all possible  $\alpha \in \mathbb{F}_q$  to see if  $f(\alpha) = 0$ . For example,  $(x^2 + x + 1)^2 = (x^2 + x + 1) \times (x^2 + x + 1)$  is reducible in  $\mathbb{F}_2[x]$  but has no root in  $\mathbb{F}_3$ .

So “roots”  $\implies$  “reducible” but “Not roots” does not imply “irreducible”.



## 4.2 Finite field with $p^m$ elements ( $p$ is a prime)

**Theorem 7.** Let  $E(x)$  be an irreducible polynomial of degree  $m \geq 2$ ,  $m \in \mathbb{F}_p[x]$  for prime  $p$ . The set of polynomials in  $\mathbb{F}_p[x]$  modulo  $E(x)$  denoted by  $\mathbb{F}_{p^m} = \mathbb{F}_p[x]/(E(x))$  is a finite field with  $p^m$  elements.

Note that  $\mathbb{F}_{p^m}$  is a finite field. It is a set of polynomials of degree less than  $m$  modulo  $E(x)$ . It is a set of polynomials of degree less than  $m$  with coefficients in  $\mathbb{F}_p$ . It is because anything higher than  $m$  can be reduced by  $E(x)$ .

- Addition:  $f(x) + g(x) = (f(x) + g(x)) \pmod{E(x)}$  is just plain addition in  $\mathbb{F}_p[x]$
- Multiplication:  $f(x) \cdot g(x) = (f(x) \cdot g(x)) \pmod{E(x)}$  is a unique remainder  $r(x)$  when  $f(x) \cdot g(x)$  is divided by  $E(x)$ .  $r(x)$  has degree less than  $E(x)$ . So  $f(x) \cdot g(x) = q(x)E(x) + r(x)$
- Additive inverse:  $f(x) \rightarrow (-f(x)) \pmod{E(x)}$
- Multiplicative inverse:  $f(x) \rightarrow g(x) = f(x)^{-1} \pmod{E(x)}$  such that  $f(x) \cdot g(x) = 1 \pmod{E(x)}$

For example: for  $p = 2$   $E(x) = 1 + x + x^2$  is irreducible in  $\mathbb{F}_2[x]$ . Then  $\mathbb{F}_{2^2} = \mathbb{F}_2[x]/(1 + x + x^2)$  is a finite field with 4 elements  $\{0, 1, x, x + 1\}$ .

The multiplicative inverse of  $\{1, x, x + 1\}$  is  $\{1, x + 1, x\}$  respectively.

**Theorem 8.** For every prime power  $p^m$ , there exists a finite field  $\mathbb{F}_{p^m}$  with  $p^m$  elements. There are no other finite fields.

When  $m = 1$ , it is a prime field  $\mathbb{F}_p$ . When  $m > 1$ , it is an extension field  $\mathbb{F}_{p^m}$ .

Note that  $\mathbb{F}_{2^2} = \{0, 1, x, x + 1\} \neq \mathbb{F}_4 = \{0, 1, 2, 3\}$ . They are different fields.  $\mathbb{F}_{2^2}$  is a field with 4 elements ( $\Sigma = 2$  with degree of at most 1).  $\mathbb{F}_4$  is a field with 4 elements ( $\Sigma = 4$  with degree of at most 0).

## 4.3 Find the irreducible polynomial

Fact:  $\forall m \in \mathbb{Z}$  and prime  $p$ ,  $\exists$  an irreducible polynomial of degree  $m$ ,  $f \in \mathbb{F}_p[x]$ .

**Theorem 9.** If  $E(x)$  is a degree  $m$  irreducible polynomial in  $\mathbb{F}_p[x]$ , then  $E(x) \mid x^{p^m} - x$ .

For example: For  $p = 2 = m$ ,  $E(x) = x^2 + x + 1$  is irreducible in  $\mathbb{F}_2[x]$ . Then  $E(x) \mid x^4 - x = x(x^3 - 1) = x(x - 1)(x^2 + x + 1)$ .

#### 4.3.1 Multiplicative group of $\mathbb{F}_{p^m}$

Fact:  $\mathbb{F}_{p^m}^*$  is a cyclic group. It has a generator  $\gamma$  such that  $\mathbb{F}_{p^m}^* = \{\gamma^0, \gamma^1, \dots, \gamma^{p^m-2}\}$ .

A good way to draw a picture in the head is to think of a circle and the circle is divided into  $p^m - 1$  equal parts. Then  $\gamma^0$  is the starting point and  $\gamma^{p^m-2}$  is the ending point. For each power of a generator, rotates the circle by a certain degree. There are a few that could be the generator, and it rotates the circle by a different degree.

#### 4.4 Polynomial Interpolation over $\mathbb{F}_{p^m}$

Make a distinction to start with:

- $\mathbb{F}_p[X] \implies f(x) = a_0 + a_1x + \dots + a_{n-1}x^{n-1}, a_i \in \mathbb{F}_p$ . So the coefficients are from  $\mathbb{F}_p$ , which are numbers only.
- $\mathbb{F}_{p^m}[X] \implies f(x) = a_0 + a_1x + \dots + a_{n-1}x^{n-1}, a_i \in \mathbb{F}_{p^m}$ . So the coefficients are from  $\mathbb{F}_{p^m}$ , which are polynomials with degree less than  $m$ .

For example, consider  $\mathbb{F}_{2^3}$

We can list out the members of the group and the corresponding multiplicative group.

$$\begin{aligned} \mathbb{F}_{2^3} &= \{ 0, 1, & x, x+1, & x^2, x^2+1, & x^2+x, x^2+x+1 \} \\ \mathbb{F}_{2^3}^* &= \{ \emptyset, \gamma^0, & \gamma^1, \gamma^2, & \gamma^3, \gamma^4, & \gamma^5, \gamma^6 \} \end{aligned}$$

Now evaluate  $f(x) = x^2 + 1$  at  $x = \gamma^3$ . we get  $f(\gamma^3) = (\gamma^3)^2 + 1 = \gamma^6 + 1 = x^2 + 1 + 1 = x^2 = \gamma^2$ . So  $f(\gamma^3) = \gamma^2$ .

Note the cyclic behavior here. And distinguish the  $x$  used as the function input and the  $x$  is used as a variable in the polynomial.

## 5 BCH Codes (Bose, Chaudhuri, Hocquenghem)

**Definition 15.** Let  $n = 2^m - 1, q = n + 1$  and  $r$  be a primitive element in  $\mathbb{F}_2^m$ . Then the BCH code  $\mathcal{C}_{BCH}(n, d)$  is defined as follows:

$$\mathcal{C}_{BCH}(n, d) = \{(c_0, c_1, \dots, c_{n-1}) : \sum_{i=0}^{n-1} c_i r^i = 0\}$$

Remark:

- $\text{BCH}(n, d) = RS_{2^m}((r^0, \dots, r^{n-1}), n, n - d + 1) \cap \mathbb{F}_2^n$
- BCH is a subset of RS code
- Distance of BCH code is  $d$
- Decoding: use Reed-Solomon decoding algorithm
- Dimension:  $\text{BCH}(n, d)$  is a linear code where  $k \geq n - \lceil \frac{d-1}{2} \rceil \log_2(n+1)$

## 5.1 Dimension of BCH code

Idea: count the number of constraints  $\{C(\gamma^j) = 0 \mid 0 < j < d\}$ .

*Proof.* Fact:  $\exists$  bijection  $\Phi : \mathbb{F}_{2^m} \rightarrow \mathbb{F}_2^m$  such that  $\Phi(0) = 0$ ,  $\Phi(a + b) = \Phi(a) + \Phi(b)$ ,  $\Phi(ab) = \Phi(a)\Phi(b)$ .

Now we have  $\Phi(C(\gamma^j)) = \Phi(\sum_{i=0}^{n-1} C_i(\gamma^j)^i) = \sum_{i=0}^{n-1} \Phi(C_i \gamma^{ij}) = \sum_{i=0}^{n-1} C_i \Phi(\gamma^{ij}) = 0$ .

wrlghaluvhbiauwh

□

## 5.2 Rate and distance of BCH code

- Rate:  $R = \frac{k}{n} = \frac{n - \lceil \frac{d-1}{2} \rceil \log_2(n+1)}{n} \approx 1 - \frac{d \log_2 n}{n} \implies \frac{d}{n} \approx \frac{2}{\log_2 n} (1 - R)$

# 6 Reed-Muller Codes

## 6.1 Multi-variate Polynomials

**Definition 16.** A multi-variate polynomial  $f(x_1, x_2, \dots, x_n)$  is a polynomial in  $n$  variables  $x_1, x_2, \dots, x_n$ .

Let  $\mathbb{F}_q[x_1, x_2, \dots, x_n]$  be the set of all multi-variate polynomials in  $n$  variables with coefficients in  $\mathbb{F}_q$ .

For example:  $f(X_1, X_2) = X_1^2 + X_1 X_2 + X_2^2$  is a multi-variate polynomial in two variables.

- Monomial: the term without coefficient. For example:  $X_1^2 X_2^3$  is a monomial.

- The total degree of a multi-variate polynomial is the sum of the degrees of all monomials.
- The degree of  $X_i$  in  $X_1^{a_1} X_2^{a_2} \cdots X_n^{a_n}$  is  $a_i$ .
- The degree of a polynomial is the maximum degree of all monomials with non-zero coefficients. For example: the degree of  $X_1^2 + X_1^2 X_2 + X_2^2$  is 3 since the monomial  $X_1^2 X_2$  has degree 3.

## 6.2 Reed-Muller Codes definition

**Definition 17.** The  $m$ -variate, Reed-Muller code of order  $r$  over  $\mathbb{F}_q$  is defined as follows:

$$RM(q, m, r) = \{f(\alpha_1), \dots, f(\alpha_{q^m}) : f \in \mathbb{F}_q[x_1, x_2, \dots, x_m]\}$$

where  $\deg(f) \leq r$ ,  $\deg_{x_i}(f) \leq q - 1$  for  $i = 1, 2, \dots, m$ , and  $\alpha_1, \alpha_2, \dots, \alpha_{q^m}$  are distinct elements in  $\mathbb{F}_q^m$ .

For example:  $q = 2, m = 2, r = 1$ . Then  $RM(2, 2, 1) = \{f(\alpha_1, \alpha_2) : f \in \mathbb{F}_2[x_1, x_2], \deg(f) \leq 1, \deg_{x_1}(f) \leq 1, \deg_{x_2}(f) \leq 1\}$

Here  $\alpha_1 = (0, 0), \alpha_2 = (0, 1), \alpha_3 = (1, 0), \alpha_4 = (1, 1)$ . So the codewords are  $(f(0, 0), f(0, 1), f(1, 0), f(1, 1))^T$ , a 4 bits vector.

All polynomials  $\rightarrow$  Codewords

$$0 \rightarrow (0, 0, 0, 0)^T$$

$$1 \rightarrow (1, 1, 1, 1)^T$$

$$x_1 \rightarrow (0, 0, 1, 1)^T$$

$$x_2 \rightarrow (0, 1, 0, 1)^T$$

$$x_1 + 1 \rightarrow (1, 1, 0, 0)^T$$

$$x_2 + 1 \rightarrow (1, 0, 1, 0)^T$$

$$x_1 + x_2 \rightarrow (0, 1, 1, 0)^T$$

$$x_1 + x_2 + 1 \rightarrow (1, 0, 0, 1)^T$$

## 6.3 Binary Reed-Muller Codes

**Definition 18.** The binary Reed-Muller code of order  $r$  is defined as follows:

$$RM(q, m, r) = RM(2, m, r)$$

### 6.3.1 Generator matrix

The generator matrix of  $RM(2, m, r)$  is a  $q^m \times 2^m$  matrix  $G$  such that  $RM(2, m, r) = \{G \cdot X : X \in \mathbb{F}_2^m\}$ .

Define it recursively, base case:

Let  $G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$  be the generator matrix of  $RM(2, 1, 1)$ .

Then  $RM(2, 1, 1) = \{G_2 \cdot X : X \in \mathbb{F}_2\}$ .

$$G_4 = \begin{bmatrix} G_2 & 0 \\ G_2 & G_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$G_8 = \begin{bmatrix} G_4 & 0 \\ G_4 & G_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Then inductively:  $G_{2^m} = \begin{bmatrix} G_{2^{m-1}} & 0 \\ G_{2^{m-1}} & G_{2^{m-1}} \end{bmatrix}$

Each column is a codeword in the generator matrix. They all correspond to a combination of the input vector  $X_1, X_2, \dots, X_m$ .

For example, for  $G_8$   $RM(2,3,3)$ :

The generator matrix contains all codewords under the degree of the code.

Note that  $RM(a, m, m)$  contains all possible codewords in the space which does not do error correction. It is discussed here to derive the subspace codes.

### 6.3.2 Properties of Binary Reed-Muller Codes

RMK: Over  $\mathbb{F}_2, \forall x \in \mathbb{F}_2, x^2 = x \iff (1^2 = 1, 0^2 = 0)$ .

So we can assume that all monomials have each variable raised to the power of at most 1.  $X_1^3 X_2^2$  can be reduced to  $X_1 X_2$ .

RMK: Any  $f \in [X_1, X_2, \dots, X_m]$  of degree  $\leq r$  can be written as a linear combination of monomials of degree  $\leq r$ .

$$f \in [X_1, X_2, \dots, X_m] = \sum_{S \subseteq [m], |S| \leq r} a_S \prod_{i \in S} X_i$$

Proposition: The dimension of  $RM(2, m, r)$  is  $\sum_{i=0}^r \binom{m}{i}$ .

Distance of  $RM(2, m, r)$ :  $d = 2^{m-r}$ .  $n = 2^m$ .

**Lemma 1. Schwartz-Zippel Lemma:** Let  $f(x_1, x_2, \dots, x_m)$  be a polynomial of degree  $d$  over  $\mathbb{F}_q$ . Let  $S \subseteq \mathbb{F}_q^m$  be a set of points. Then

$$\Pr[f(x_1, x_2, \dots, x_m) = 0, x \in S] \leq \frac{d}{|S|}$$

Do not know what this lemma is !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

### 6.3.3 Rate v.s. Distance

Consider  $RM(2, m, r)$ .

- $n = 2^m$
- $k = \sum_{i=0}^r \binom{m}{i}$
- $d = 2^{m-r}$
- Relative distance  $\delta = \frac{d}{n} = \frac{2^{m-r}}{2^m} = 2^{-r}$
- Keep  $r$  constant:  $R = \frac{k}{n} = \frac{\sum_{i=0}^r \binom{m}{i}}{2^m}$ .  $\lim_{r < m} \Rightarrow \frac{r}{m} \rightarrow 0 \approx \frac{2^{mH_2(\frac{r}{m})}}{2^m}$  where  $H_2$  is the binary entropy function.
- $H_2(x) = -x \log_2(x) - (1-x) \log_2(1-x)$ ,  $x \in [0, 1]$
- So for distance to be large, the rate has to approach 0. Needs to do a trade-off between rate and distance.

## 6.4 Low degree Reed-Muller Codes

**Definition 19.** The  $m$ -variate, Reed-Muller code of order  $r$  over  $\mathbb{F}_q$  is defined as follows:

$$RM(q, m, r) = \{f(\alpha_1), \dots, f(\alpha_{q^m}) : f \in \mathbb{F}_q[x_1, x_2, \dots, x_m], \deg(f) \subseteq r\}$$

RMK: When  $r < q$ ,  $\deg_{x_i}(f) \leq \deg(f) \leq r \leq q$ . We can drop the condition  $\deg_{x_i}(f) \leq q-1$  since it is redundant.

- Dimension of  $RM(q, m, r)$  is  $\binom{m+r}{m}$

- Distance of  $RM(q, m, r)$  is  $d \geq (q - r)q^{m-1}$

If we fix  $m = 2$ , then  $q = \sqrt{n} \implies \delta = \frac{d}{n} = \frac{q-r}{q} = 1 - \frac{r}{\sqrt{n}} \implies R = \frac{\binom{2+r}{r}}{n} = \frac{(r+2)(r+1)}{2n} \geq \frac{r^2}{2n} = \frac{(1-\delta)^2}{2}$

Note that we can have a positive rate and distance at the same time now which is a large improvement.

#### 6.4.1 Proof of Dimension of Low Degree Reed-Muller Codes

*Proof.* Then  $RM(q, m, r)$  can be generated by monomial basis  $\{X_1^{d_1} X_2^{d_2} \cdots X_m^{d_m} : \sum_{i=1}^m d_i \leq r\}$ .

The dimension of the code is the same as the size of  $D = \{d_1, d_2, \dots, d_m : \sum_{i=1}^m d_i \leq r\}$ .  $k = |D| = \binom{m+r}{m}$ .

$|D| = \sum_{j=0}^k \text{number of solutions to } \sum_{i=1}^m d_i = j$ .

$$j = 1 \implies m$$

$$j = 2 \implies X_i^2 \vee X_i X_k \implies \binom{m}{2} + m = \binom{m+1}{2}$$

$$\begin{aligned} |D_j| &= \sum_{k=1}^m |D_{jk}| \\ &= \sum_{k=1}^m \binom{m}{k} \binom{j-1}{k-1} \\ |D_{jk}| &= \binom{m}{k} \binom{j-1}{k-1} \end{aligned}$$

To be added!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

#### 6.4.2 Proof of Distance of Low Degree Reed-Muller Codes

**Lemma 2.** Let  $f$  be a non-zero polynomial with  $\deg(f) \leq r$ . Then the fraction of zeros of  $f$  is at most  $\frac{r}{q}$ .

RMK: This implies that ... some equations here.

RMK: If  $m = 1$  here, then  $RM(q, 1, r)$  is a Reed-Solomon code. The lemma says  $\{\alpha \in \mathbb{F}_q : f(\alpha) = 0\} \leq r$  which is the same as the Singleton bound. It means multi-variable poly of degree  $r$  has no more than  $r$  roots.

RMK: The lemma is equivalent as saying when  $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_m)$  chosen uniformly from  $\mathbb{F}_q^m$ ,  $\Pr[f(\alpha) = 0] \leq \frac{r}{q}$ .

*Proof.* Proof by induction:

Base case:  $m = 1$ .  $f(x) = a_0 + a_1x + \dots + a_rx^r$ .  $f(x) = 0$  has at most  $r$  roots as shown in the remark.  $\square$

where is proof?  $\square$

## 7 Stochastic noise channel

**Definition 20.** A stochastic noise channel is a channel where the output is a random variable  $Y$  given the input  $X$ . The probability of  $Y$  given  $X$  is  $P(Y|X)$ .

Noise can be:

- flipping up to  $t$  bits
- erasing up to  $e$  bits

We want to decode the message correctly with a high probability no matter what the noise is.

Shannon: what if the noise is stochastic?

### 7.1 Some common stochastic noise channels

Common assumption: Noise is independent and identically distributed (i.i.d.) over each transmission

- Binary symmetric channel (BSC):  $P(Y = 1|X = 0) = P(Y = 0|X = 1) = p$  if  $X$  is the input  $Y$  is the output. Whether you send 0 or 1, there is a probability  $p$  that it will be flipped.  
 $\forall i \in [n], Y_i = X_i \oplus Z_i$  where  $Z_i$  is a random variable with  $P(Z_i = 1) = p, P(Z_i = 0) = 1 - p$ . The noise  $Z$  is independent of the input  $X$ .
- Binary erasure channel (BEC):  $P(Y = 1|X = 1) = P(Y = 0|X = 0) = 1 - \epsilon, P(Y = ?|X = 0) = P(Y = ?|X = 1) = \epsilon$ . Here  $?$  means the bit is erased.



$\forall i \in [n], Y_i = X_i$  with probability  $1 - \epsilon$ ,  $Y_i = ?$  with probability  $\epsilon$ .

iid noise assumption  $\implies P(Y^n|X^n) = \prod_{i=1}^n P(Y_i|X_i)$

RMK: A stochastic noise channel is fully characterized by the channel transition probability  $P(Y|X)$ .

- Additive white Gaussian noise (AWGN):  $Y = X + N$  where  $N \sim \mathcal{N}(0, \sigma^2)$ 
  - Map  $\{0, 1\} \rightarrow \{-1, 1\}$
  - Let  $Z_i \sim \mathcal{N}(0, \sigma^2)$  be the noise. The Gaussian noise with zero mean and variance  $\sigma^2$ , independent of the input  $X$ .
  - Then  $Y_i = X_i + Z_i, \forall i \in [n]$ .  $P(Y|X)$  is the Gaussian distribution.

$$P(Y|X) = P(Z = Y - X) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(Y-X)^2}{2\sigma^2}}$$

. This is the probability density function of the Gaussian distribution.

- Given  $X_i = 1, Y_i \sim \mathcal{N}(1, \sigma^2)$  Shift the mean from 0 to 1.
- Given  $X_i = -1, Y_i \sim \mathcal{N}(-1, \sigma^2)$  Shift the mean from 0 to -1.

## 7.2 New performance criteria for stochastic noise channel

Since the noise is stochastic, we cannot guarantee that the message will have up to  $t$  errors or  $e$  erasures. We are motivated to define the “probability of error” and the “probability of erasure”.

probability of error  $P_e = \Pr[\hat{X} \neq X]$

probability of erasure  $P_\epsilon = \Pr[\hat{X} = ?]$

We want to design a code such that  $P_e$  and  $P_\epsilon$  are small.

### 7.2.1 Tradeoff between $R$ V.S. $P_e$ and $P_\epsilon$ for repetition code

$$\text{ENC: } \begin{cases} u \rightarrow x^n = u \otimes 1^n \\ 0 \rightarrow 0^n \\ 1 \rightarrow 1^n \end{cases}$$

### 7.2.2 maximum likelihood decoding

Given  $Y^n$ , find  $\hat{X}^n$  that maximizes  $P(Y^n|\hat{X}^n)$ . If the probability is equal, break the tie arbitrarily and uniformly.

RMK: Bayesian inference: Under the uniform message assumption, the **optimal** decoding rule (in the sense of minimizing the probability of error) for stochastic noise channel is the following:

$$\hat{X}^n = \arg \max_{x^n \in \mathcal{C}} P(Y^n|X^n)$$

break the tie arbitrarily and uniformly.

$\hat{u}^k$  is the message corresponding to  $\hat{X}^n$ .

### 7.3 Coding over BSC(p)

**Definition 21.** A code  $\mathcal{C}$  is said to be **good** for the BSC(p) if the probability of error  $P_e$  is small.

### 7.4 Hamming code over BSC(p)

Let  $p$  be the probability of bit flip. Since the distance of the Hamming code is 3, it can correct 1 error. Probability of error  $P_e$  that can be corrected is:

$$P_e = 1 - (1 - p)^7 - \binom{7}{1}(1 - p)^6 p$$

syndrome decoding for hamming code: