Introduction

Information theory studies reliable information transmission over an unreliable channel. Coding theory is a subfield of information theory, which studies methods that allow for reliable information transmission.

Coding theory is also used in:

1. Cryptography (secret sharing, private information retrieval, McEliece cryptosystem)
2. Theoretical computer science (probabilistically checkable proofs, list decoding, local decodability)
3. Signal processing (compressed sensing)
4. Networking (network coding)
5. Biology (study of DNA)

Examples in Real Life

Coding theory primary goal is error correction. Some real life examples of systems and devices that use error-correcting codes:

1. Wired and wireless communications (wi-fi, mobile phones, etc.)
2. Memory cards and flash drives
3. Hard drives
4. CD and DVD

In all of the above examples the data is transmitted (or stored) through (in) unreliable medium. This typically causes some errors, which are to be corrected by using error-correction mechanisms.

Communication Model of Shannon

Communications model was proposed by Claude Shannon in 1948. The communications system consists of several components. Each component has an input and an output. The diagram below shows the connection between the different components.
Information is generated by source and is transferred over the channel. It consists of bits or symbols of information. It is represented in the figure by \( \mathbf{x} \), where \( \mathbf{x} = (x_1, x_2, ..., x_k) \). This \( \mathbf{x} \) serves as an input to the Encoder.

The output of the Encoder is \( \mathbf{c} = (c_1, c_2, ..., c_n) \), a vector over an alphabets \( \Sigma_{in} \). It also serves as the channel input. The channel output is \( \mathbf{y} = (y_1, y_2, ..., y_n) \), a vector over an alphabet \( \Sigma_{out} \).

The Decoder attempts to restore \( \mathbf{c} \) from \( \mathbf{y} \) (in some variations of this model, the decoder also attempts to recover \( \mathbf{x} \)).

The above alphabets can also be continuous, but for the scope of this course we will focus on discrete alphabets.

### Channels

A channel is defined by the triple \((\Sigma_{in}, \Sigma_{out}, \text{Prob})\). Here, \( \Sigma_{in} \) and \( \Sigma_{out} \) are input and output alphabets, respectively. Alphabet can be either discrete (for example, binary) or continuous.

Probability function \( \text{Prob} : \Sigma_{in} \times \Sigma_{out} \to [0, 1] \) is defined on pairs of symbols as

\[
\text{Prob}(a, b) = \Pr(b \text{ received} \mid a \text{ transmitted}),
\]

where \( \Pr(b \mid a) \) denotes conditional probability.

In this course, we focus on memoryless channels, where the symbols are independent of each other. Discrete memoryless channels have discrete input and output alphabets, and the symbols are independent of each other.

### Binary Symmetric Channel (BSC)

Binary symmetric channel is a binary channel, which erases each bit with probability \( p \in [0, 1] \). It is defined as follows: \( \Sigma_{in} = \Sigma_{out} = \{0, 1\} \), and

\[
\begin{align*}
\Pr(b = 1 \mid a = 1) &= \Pr(b = 0 \mid a = 0) = 1 - p \\
\Pr(b = 1 \mid a = 0) &= \Pr(b = 0 \mid a = 1) = p
\end{align*}
\]

where \( p \in [0, 1] \) is a crossover probability, \( a \in \{0, 1\} \) represents a transmitted symbol and \( b \in \{0, 1\} \) represents a received symbol.

If a received bit is different from a transmitted bit, then there is an error. This happens with the probability \( p \). Binary symmetric channel with crossover probability \( p \) will also be denoted as BSC\((p)\).
Consider BSC(\(p\)): if the value of \(p\) is close to 0 then the errors are rare, and the received sequence will be rather similar to the transmitted sequence. If \(p\) is close to 1, then with high probability all bits were flipped, and so the transmitted sequence will be similar to a sequence obtained by reversing all received bits. However, if \(p\) is close to \(\frac{1}{2}\), the received bits will look rather random.

**Binary Erasure Channel (BEC)**

Binary erasure channel is a binary-input channel, which erases each bit with probability \(p \in [0, 1]\). It is defined as follows: \(\Sigma_{in} = \{0, 1\}, \Sigma_{out} = \{0, 1, \epsilon\}\), and

\[
\begin{align*}
\Pr(b = 1 | a = 1) &= \Pr(b = 0 | a = 0) = 1 - p \\
\Pr(b = \epsilon | a = 1) &= \Pr(b = \epsilon | a = 0) = p \\
\Pr(b = 1 | a = 0) &= \Pr(b = 0 | a = 1) = 0
\end{align*}
\]

where \(p \in [0, 1]\) is an erasure probability.

If \(p\) is close to 1 then nearly all bits are erased. If \(p\) is close to 0, then most of the symbols are intact.

**Other channels**

Other important channels include \(q\)-ary symmetric channel, \(q\)-ary erasure channel and additive white Gaussian noise (AWGN) channel.
Encoder

In block coding, $\overline{x} = (x_1, x_2, ..., x_k)$ is an information word of length $k$, which is encoded into a codeword $\overline{c} = (c_1, c_2, ..., c_n)$ of length $n$. In the sequel, we focus on the block codes.

Definition: An $(n, M)$-code $\mathcal{C}$ over an alphabet $F$ is a set of $M > 0$ vectors (called codewords) of length $n$ over $F$.

Two different inputs are mapped onto two different outputs, otherwise the decoding will not be possible.

Code parameters

- Length: $n$
- Size or cardinality: $M$
- Dimension: $k = \log|F| M$
- Rate: $r = \frac{k}{n}$

Hamming Distance

Hamming distance can be used to measure the difference between two vectors.

Definition: Let $\overline{x} = (x_1, x_2, ..., x_n)$ and $\overline{y} = (y_1, y_2, ..., y_n)$ be two vectors. The Hamming distance between $\overline{x}$ and $\overline{y}$, $d(\overline{x}, \overline{y})$, is defined as the number of coordinates which are pairwise different in $\overline{x}$ and $\overline{y}$, i.e.

$$d(\overline{x}, \overline{y}) \triangleq \left| \{i \mid x_i \neq y_i \} \right|.$$

Example: Let $\overline{x} = (0, 0, 1)$ and $\overline{y} = \{1, 1, 1\}$. The Hamming distance $d(\overline{x}, \overline{y})$ is 2.

Metric

Let $F$ be some alphabet. Denote by $F^n$ a set of all vectors of length $n$ over the alphabet $F$. Metric $d$ over $F^n$ is a function $d: F^n \times F^n \to \mathbb{R}$ that satisfies the following three conditions.

1. Non-negativity: $d(\overline{x}, \overline{y}) \geq 0$. Moreover, $d(\overline{x}, \overline{y}) = 0$ if and only if $\overline{x} = \overline{y}$.
2. Symmetry: $d(\overline{x}, \overline{y}) = d(\overline{y}, \overline{x})$.
3. Triangle inequality: For any three vectors $\overline{x}, \overline{y}, \overline{z} \in F^n$,

$$d(\overline{x}, \overline{z}) + d(\overline{z}, \overline{y}) \geq d(\overline{x}, \overline{y}).$$
It can be easily checked that Hamming distance is a metric.

**Example:** Take $F = \{0, 1\}$. Assume that addition modulo 2 is used. Then, for any $\overline{x}, \overline{y} \in F^n$:

$$d(\overline{x}, \overline{y}) = w(\overline{x} + \overline{y}) .$$

In other words, the Hamming distance between $\overline{x}$ and $\overline{y}$ is equal to the Hamming weight (the number of non-zero coordinates) in $\overline{x} + \overline{y}$. 
Minimum Distance

Let $F$ be an alphabet. Recall the parameters of an $(n, M)$ block code over the alphabet $F$:

- $n$ is a code length;
- $M$ is a cardinality;
- $K = \log_{|F|} M$ is a dimension;
- $r = \frac{K}{n}$ is a code rate.

**Definition:** The Hamming Distance $d(\bar{x}, \bar{y})$ between two vectors $\bar{x} \in F^n$ and $\bar{y} \in F^n$ is given by

$$d(\bar{x}, \bar{y}) = \left| \{ i \mid x_i \neq y_i \} \right|.$$ 

**Definition:** The minimum distance of the code $C$ is

$$d \triangleq \min_{\bar{c}_1, \bar{c}_2 \in C, \bar{c}_1 \neq \bar{c}_2} \left\{ d(\bar{c}_1, \bar{c}_2) \right\}.$$ 

Informally, the minimum distance indicates how far apart the codewords in the code are.

**Notation:** A block code of length $n$, cardinality $M$ and minimum distance $d$ will be denoted as an $(n, M, d)$ code.

**Example:** The binary (3, 2, 3) repetition code. Here $F = \{0, 1\}$.

$$C = \{(0, 0, 0), (1, 1, 1)\} \subseteq F^3.$$ 

Parameters:

- $n = 3, M = 2, d = 3$;
- $K = \log_2 2 = 1$,
- $r = \frac{1}{3}$.

**Example:** The binary (3, 4, 2) parity code. Again, $F = \{0, 1\}$.

$$C = \{(0, 0, 0), (0, 1, 1), (1, 1, 0), (1, 0, 1)\} \subseteq F^3.$$
Parameters:

- \(n = 3, M = 4, d = 3;\)
- \(K = \log_2 4 = 2;\)
- \(r = \frac{2}{3}.\)

Decoding

**Definition:** A *decoder* for the code \(C\) is a function \(D : \Sigma_{\text{out}}^n \to C.\)

We say that a *decoding error* has occurred if \(\bar{c}\) was transmitted, \(\bar{y}\) was received, and \(D(\bar{y}) \neq \bar{c}\).

**Decoding error probability**

Define the probability of a decoding error, \(P_{\text{err}}\), as follows:

\[
P_{\text{err}} = \max_{\bar{c} \in C} \{ P_{\text{err}}(\bar{c}) \},
\]

where

\[
P_{\text{err}}(\bar{c}) = \sum_{\bar{y} : D(\bar{y}) \neq \bar{c}} \text{Prob}(\bar{y} \text{ received } | \bar{c} \text{ transmitted}).
\]

The goal is to make \(P_{\text{err}}\) as small as possible.

**Example**

Let \(C\) be the binary \((3, 2, 3)\) repetition code, used over BSC(\(p\)), and \(F = \{0, 1\}\). Define the encoder \(E : F \to F^3\) as follows:

\[
\begin{align*}
E(0) &= (0, 0, 0) \\
E(1) &= (1, 1, 1)
\end{align*}
\]

We can define the decoding function \(D : F^3 \to C\), where

\[
\begin{align*}
D((0, 0, 0)) &= D((0, 0, 1)) = D((0, 1, 0)) = D((1, 0, 0)) = (0, 0, 0) \\
D((1, 1, 1)) &= D((1, 1, 0)) = D((1, 0, 1)) = D((0, 1, 1)) = (1, 1, 1)
\end{align*}
\]

The decoding rule as above is a majority vote: if there are more 0’s than 1’s, then the word is decoded into all-zero word. If there are more 1’s then 0’s, then it is decoded into all-one word.

Assume that the binary \((3, 2, 3)\) repetition code is used over the BSC(\(p\)) channel, and the word \(\bar{c} = (0, 0, 0)\) is transmitted. The decoder \(D\) as above fails only if the received word contains at least
two 1’s. Such event corresponds to the situation, when there were two or three bit errors in the transmitted word. Then, the probability of the decoding error is given by:

\[ P_{\text{err}}(\bar{c}) = p^3 + 3p^2(1 - p) = 3p^2 - 2p^3. \]

Due to the symmetry, the probability of the decoding error when \( \bar{c} = (1, 1, 1) \) is transmitted can be obtained in the same way. Therefore,

\[ P_{\text{err}} = P_{\text{err}}((0, 0, 0)) = P_{\text{err}}((1, 1, 1)) = 3p^2 - 2p^3. \]

Note that without coding, the error probability in BSC(\( p \)) is \( p \). Let’s compare the probability \( P_{\text{err}} \) with \( p \). We have that for \( 0 < p < \frac{1}{2} \),

\[ 3p^2 - 2p^3 - p = -p(1 - p)(1 - 2p) < 0. \]

Therefore, for \( 0 < p < \frac{1}{2} \), we have \( 3p^2 - 2p^3 < p \). This means that if the repetition code is employed, then the probability of the decoding error is smaller, when compared to the case where no coding is used. However, in this example, we had to use a code of rate \( \frac{1}{3} \) instead of rate of 1 if no coding is used.

Decoding techniques

Given the received word \( \bar{y} \in F^n \), how do we decide what codeword \( \bar{c} \) was transmitted? There are several ways to make this decision.

**Nearest neighbour decoding.** In this method, we select a codeword which has the minimum Hamming distance from the received word. The decision rule is:

\[ D(\bar{y}) = \bar{z} \quad \text{such that} \quad \bar{z} = \arg \min_{\bar{c} \in C} \{d(\bar{y}, \bar{c})\}. \]

**Maximum-likelihood (ML) decoding.** In this method, we select a codeword which maximizes the following conditional probability:

\[ D(\bar{y}) = \bar{z} \quad \text{such that} \quad \bar{z} = \arg \max_{\bar{c} \in C} \{ \Prob(\bar{y} \text{ received} \mid \bar{c} \text{ transmitted}) \}. \]

The nearest-neighbour decoding is relatively simple. It depends only on the structure of the code, and it does not depend on the underlying communication channel. ML-decoding is often considered as the most accurate decoding technique. The following theorem shows that the two techniques are equivalent when used over the binary symmetric channel.

**Theorem.** Consider the binary symmetric channel with the crossover probability \( 0 < p < \frac{1}{2} \). Then the ML-decoding is equivalent to the nearest-neighbour decoding.

**Proof.** Assume that the codeword \( \bar{c} = (c_1, c_2, \cdots, c_n) \in C \) is transmitted over the BSC(\( p \)), and the word \( \bar{y} = (y_1, y_2, \cdots, y_n) \in F^n \) is received.
First, consider the bit $i$ in the transmitted and the received word. Note that from the definition of BSC($p$),
\[
\Pr(y_i \text{ received} | c_i \text{ transmitted}) = \begin{cases} 
p & \text{if } y_i \neq c_i \\
1 - p & \text{if } y_i = c_i
\end{cases}.
\]
There are exactly $d(\bar{y}, \bar{c})$ pairwise different bits in $\bar{y}$ and $\bar{c}$. We have,
\[
\Pr(\bar{y} \text{ received} | \bar{c} \text{ transmitted}) = \prod_{i=1}^{n} \Pr(y_i \text{ received} | c_i \text{ transmitted}) = (1 - p)^{n - d(\bar{y}, \bar{c})} \cdot p^{d(\bar{y}, \bar{c})},
\]
where the first transition is due to the memoryless character of the channel (the values of the different bits are independent of each other), and the second transition is obtained by counting how many pairs of bits are different and how many are the same.

In the ML-decoding, we aim to maximize the above probability. If $0 < p < \frac{1}{2}$ then $0 < \frac{p}{1-p} < 1$, and so the maximum of the probability is obtained when the power $d(\bar{y}, \bar{c})$ is as small as possible. However, this is exactly the definition of the nearest-neighbour decoding. We conclude that the ML-decoding is equivalent to the nearest-neighbour decoding. \hfill \Box

**Binary Entropy Function**

The binary entropy function $H : [0, 1] \to [0, 1]$ is defined as
\[
H(x) = -x \log_2 x - (1 - x) \log_2(1 - x).
\]

![Figure 1: The binary entropy function $H(x)$.
](image-url)

It can be seen that $H(0) = H(1) = 0$, $H(\frac{1}{2}) = 1$, and $H$ is concave. Denote by $S$ the BSC($p$). The *capacity* of $S$ is given by $\text{Cap}(S) = 1 - H(p)$.
Observe that the $\text{Cap}(S) = 1$ when $p = 0$ or $p = 1$. By contrast, $\text{Cap}(S) = 0$ when $p = \frac{1}{2}$. This is in agreement with the intuition that for the most reliable channels $p$ is either close to 0 or 1, and the worst possible choice is $p = \frac{1}{2}$.

For the binary erasure channel, $\text{BEC}(p)$, the capacity is $1 - p$.

**Theorem** (Shannon’s Coding Theorem for the BSC).

Let $S$ be the BSC($p$) and let $R$ be a real number, $0 \leq R \leq \text{Cap}(S)$. Then there exists an infinite sequence of $(n_i, M_i)$ block codes over $F = \{0, 1\}$, $i = 0, 1, 2, \ldots$, such that $\frac{\log_2 M_i}{n_i} \geq R$, and for the ML-decoding, the decoding error probability $P_{\text{err}}$ approaches 0 as $i \to \infty$.

**Theorem** (Shannon’s Converse Coding Theorem for the BSC).

Let $S$ be the BSC($p$) and let $R$ be a real number, $R > \text{Cap}(S)$. Then, for any infinite sequence of $(n_i, M_i)$ block codes over $F = \{0, 1\}$, $i = 0, 1, 2, \ldots$, such that $\frac{\log_2 M_i}{n_i} \geq R$, and for any decoding scheme, the decoding error probability $P_{\text{err}}$ approaches 1 as $i \to \infty$. 

Figure 2: The capacity of the BSC($p$) as a function of $p$. 
Error correction

Let $\mathbb{F} = (F, +, \cdot)$ be a finite field, and $C$ be a code of length $n$ over $\mathbb{F}$. In the sequel, we assume that $\bar{c} = (c_1, c_2, ..., c_n) \in C$ is the transmitted codeword, and $\bar{y} = (y_1, y_2, ..., y_n)$ is the received word.

Definitions:

1. We say that there is an error in coordinate $i \in \{1, 2, ..., n\}$ if $y_i \neq c_i$.

2. The number of errors in $\bar{y}$ is the number of such coordinates, or more formally

$$| \{ i : y_i \neq c_i, \ i = 1, 2, \cdots, n \} | .$$

Theorem. Let $C$ be an $(n, M, d)$ code over $\mathbb{F}$. There exists a decoder that corrects any pattern of $\floor{\frac{d-1}{2}}$ errors.

Note: The theorem does not say anything about the decoder, – it can be totally inefficient.

Proof: Let $d(\bar{y}, \bar{c}) \leq \floor{\frac{d-1}{2}}$ and take $D$ to be the nearest-neighbour decoder for the code $C$.

Assume that the decoder outputs $\bar{z} = D(\bar{y}) \in C$ instead of expected $\bar{c} \in C$, $\bar{z} \neq \bar{c}$. By the definition of the nearest-neighbour decoder,

$$d(\bar{y}, \bar{z}) \leq d(\bar{y}, \bar{c}) \leq \floor{\frac{d-1}{2}} .$$

Otherwise, $d(\bar{y}, \bar{z}) > d(\bar{y}, \bar{c})$, and thus the nearest-neighbour decoder would output $\bar{c}$ instead of $\bar{z}$.

Due to the triangular inequality and symmetry,

$$d(\bar{z}, \bar{c}) \leq d(\bar{y}, \bar{z}) + d(\bar{y}, \bar{c}) ,$$

and from statement (1),

$$d(\bar{z}, \bar{c}) \leq d(\bar{y}, \bar{z}) + d(\bar{y}, \bar{c}) \leq \floor{\frac{d-1}{2}} + \floor{\frac{d-1}{2}} \leq d - 1 .$$

On the other hand, the minimum distance of the code $C$ is $d$. Therefore,

$$d \leq d(\bar{z}, \bar{c}) .$$
By combining this with (2), we obtain that
\[ d \leq d(\bar{z}, \bar{c}) \leq d - 1. \]
We have reached a contradiction. This means that our assumption that the decoder outputs \( \bar{z} \) instead of \( \bar{c} \) was incorrect. Therefore, there exists a decoder \( D \) that corrects any pattern of \( \left\lfloor \frac{d-1}{2} \right\rfloor \) errors.

\( \square \)

**Example.** Take the binary \((3, 2, 3)\) repetition code. Its minimum distance is \( d = 3 \). According to the previous theorem, this code can correct \( \left\lfloor \frac{d-1}{2} \right\rfloor = 1 \) error.

**Definition.** A *sphere* of radius \( t \) around the vector \( \bar{c} \in F^n \) is a set of vectors
\[ B_t(\bar{c}) = \{ \bar{x} : d(\bar{x}, \bar{c}) \leq t \} \subseteq F^n. \]

We can think of \( B_t(\bar{c}) \) as a sphere in the space \( F^n \) that has a center in the codeword \( \bar{c} \) and radius \( t \).

For illustrative purposes, consider the \((n, M, d)\) code \( C, t = \left\lfloor \frac{d-1}{2} \right\rfloor \) and codewords \( c_1, c_2, c_3 \in C \) that are centers of spheres.

![Figure 1: Spheres in space \( F^n \).](image)

The spheres in this example do not overlap. To see that, assume that there exists a word \( \bar{y} \), which is located in the intersection of two spheres: one sphere is centered at the codeword \( \bar{c}_1 \), and another is centered at the codeword \( \bar{c}_2 \). We obtain
\[ d(\bar{c}_1, \bar{c}_2) \leq d(\bar{c}_1, \bar{y}) + d(\bar{y}, \bar{c}_2) \leq \left\lfloor \frac{d-1}{2} \right\rfloor + \left\lfloor \frac{d-1}{2} \right\rfloor \leq d - 1. \]
Since the minimum distance of $C$ is $d$, we have
\[ d \leq d(\bar{c}_1, \bar{c}_2). \]
This is a contradiction to (3). Therefore, the spheres do not intersect.

**Definition.** *Error detection* is a notification that some errors occurred, without attempting to correct them.

We modify a decoder to detect errors.

\[
\mathcal{D} : F^n \mapsto (C \cup \{ 'E' \})
\]  \hspace{1cm} (4)

where the output 'E' represents an occurrence of error.

**Theorem.** Let $C$ be an $(n, M, d)$ code over $F$. Then there is a decoder as in (4) that detects any pattern of up to $d - 1$ errors.

**Proof:** Define the decoder $\mathcal{D}$ as follows:

\[
\mathcal{D}(\bar{y}) = \begin{cases} 
\bar{y} & \text{if } \bar{y} \in C \\
'E' & \text{otherwise}
\end{cases}
\]

The detection will fail if $\bar{y} \in C$ and $\bar{y} \neq \bar{c}$. However, $d(\bar{y}, \bar{c}) \geq d$. Therefore, detection will fail only if there are $d$ or more errors. \hfill \Box

**Example.** Take the binary $(3, 2, 3)$ repetition code $C$. According to the theorem, it can detect up to 2 errors.

Assume that $\bar{c} = (000) \in C$ was transmitted. By having two arbitrary bit errors, we obtain, for example, $\bar{y} = (101)$. Aforementioned decoder outputs 'E' because $\bar{y}$ is not a codeword.

**Example.** Take the binary $(n, 2, n)$ repetition code $C$. We have
\[
C = \left\{ \left( \underbrace{00 \cdots 0}_n, \underbrace{11 \cdots 1}_n \right) \right\}
\]

This code can correct $\left\lfloor \frac{n-1}{2} \right\rfloor$ errors or detect $n - 1$ errors. In order to perform the correction, we can use a simple "majority voting". In other words, if we receive a word with more 0’s than 1’s, then we correct it to $(00 \cdots 0)$. If we receive a word with more 1’s than 0’s, then we correct it to $(11 \cdots 1)$. If $n$ is even and there are as many 0’s as there are 1’s, then we can choose the outcome of correction randomly.

**Example.** Take the binary $(3, 4, 2)$ parity code $C$. In this case,
\[
C = \{ (000), (101), (011), (110) \}
\]
and
\[
\left\lfloor \frac{d-1}{2} \right\rfloor = \left\lfloor \frac{2-1}{2} \right\rfloor = 0.
\]
This means that the code cannot correct any errors, but it can detect one error.
Erasure correction

Assume that
\[ \bar{c} = (c_1, c_2, ..., c_n) \in F^n \] is transmitted,
\[ \bar{y} = (y_1, y_2, ..., y_n) \in (F \cup \{\varepsilon\})^n \] is received,
where \( \varepsilon \) denotes a symbol erasure. Suppose there are only erasures in the received word, but no errors.

Figure 2: Binary erasure channel, BEC(\( p \))

**Theorem.** Let \( C \) be an \((n, M, d)\) code over \( F \). There is a decoder that corrects any pattern of up to \( d - 1 \) erasures.

**Proof:** Consider a decoder
\[ D : (F \cup \{\varepsilon\})^n \mapsto C. \]
The decoder is defined as follows: \( D(\bar{y}) = \bar{z} \in C \) if \( y \) agrees with exactly one \( \bar{z} \in C \) on all its non-erased coordinates.

Now, suppose that the number of erasures in \( \bar{y} \) is \( \leq d - 1 \). We know that there is at least one codeword that agrees with \( \bar{y} \), that is the transmitted codeword \( \bar{c} \).

Assume that there are more than one codeword that agree with \( \bar{y} \). Take two of them, say, \( \bar{z}_1 \) and \( \bar{z}_2 \) that both agree with \( \bar{y} \) on all its non-erased coordinates. Then they can differ only at coordinates which correspond to erasures in \( \bar{y} \). This means that
\[ d(\bar{z}_1, \bar{z}_2) \leq d - 1. \]

However, \( d \) is the minimum distance of \( C \). Therefore, we obtain a contradiction. In other words, there is only one codeword that agrees with \( \bar{y} \). It also means that \( D \) is the decoder with the requested properties.

**Theorem.** Let \( C \) be an \((n, M, d)\) code. Then there exists a decoder that corrects any pattern of \( \rho \) erasures and \( \tau \) errors whenever \( \rho + 2\tau \leq d - 1 \).
Linear codes

Definition: Let $F = (F, +, \cdot)$ be a finite field. An $(n, M, d)$ code over $F$ is called linear if $C$ is a nonempty linear vector subspace of $F^n$.

Reminder: The following conditions must be met in order for $C$ to be a linear subspace of $F^n$.

(i) if $\bar{c}_1, \bar{c}_2 \in C$ then $\bar{c}_1 + \bar{c}_2 \in C$,

(ii) if $\bar{c} \in C$, $\alpha \in F$ then $\alpha \cdot \bar{c} \in C$.

Note: The dimension of $C$ as a linear subspace is $k = \log_{|F|} M$, which coincides with the definition of the dimension of the code in Lecture 1.

Notation: If $C$ is a linear code of length $n$, dimension $k$ and minimum distance $d$, we say that is an \([n, k, d]\) code over $F$.

Example: Consider the $(3, 4, 2)$ binary parity code over $F_2$,

$$C = \{(000), (011), (101), (110)\}.$$  

This code is a vector subspace in $(F_2)^3$. The following claims can be easily shown.

- The dimension of $C$ is $k = \log_{|F|} M = \log_2 4 = 2$.
- One possible basis for $C$ is $\{ (011), (101) \}$. To prove this, it suffices to show that all the codewords can be expressed as linear combinations of the basis elements:

$$
\begin{align*}
(011) &= 1 \cdot (011) + 0 \cdot (101) \\
(101) &= 0 \cdot (011) + 1 \cdot (101) \\
(000) &= 0 \cdot (011) + 0 \cdot (101) \\
(110) &= 1 \cdot (011) + 1 \cdot (101) .
\end{align*}
$$
Definition. A generator matrix of a linear \([n, k, d]\)-code is a \(k \times n\) matrix whose rows form a basis of the code.

Note. We usually denote a generator matrix of a code with a letter \(G\).

Example. Let
\[
G = \begin{pmatrix}
1 & 0 & 1 \\
0 & 1 & 1
\end{pmatrix}
\]
be a generator matrix of the \((3, 2, 3)\)-parity code \(C\). For example,
\[
\hat{G} = \begin{pmatrix}
1 & 0 & 1 \\
1 & 1 & 0
\end{pmatrix}
\]
is another generator matrix of \(C\). Typically, code has many generator matrices.

Example. A generator matrix of \([n, n-1, 2]\)-parity code over a finite field \(F\)
\[
\begin{pmatrix}
1 & 0 & 0 & \ldots & 0 & -1 \\
0 & 1 & 0 & \ldots & 0 & -1 \\
0 & 0 & 1 & \ldots & 0 & -1 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \ldots & 1 & -1
\end{pmatrix}
\]
\(n-1\) rows
\(n\) columns

A generator matrix of \([n, n-1, 2]\) can be described as follows:

1. The submatrix consisting of the first \(n-1\) columns in \((n-1) \times (n-1)\) identity matrix.
2. The last column has value \(-1\) in all positions.

The parity code is defined as a set of all vectors, such that the sum of their entries over \(F\) is zero.

Example. A binary repetition code of length 3:
\[
C = \{(000), (111)\}.
\]
It is a vector space over \(F_2\) spanned by a single vector \((111)\), denoted as \((111)\). Then, \(C\) is a linear \([3, 1, 3]\)-code with a generator matrix
\[
G = \begin{pmatrix}
1 & 1 & 1
\end{pmatrix}
\]
Example. The \([n, 1, n]\) repetition code of length \(n\) over a finite field \(\mathbb{F}\) is defined as the code, whose generator matrix is

\[
G = \begin{pmatrix}
1 & 1 & \cdots & 1
\end{pmatrix}.
\]

Theorem. Let \(\mathcal{C}\) be a linear \([n, k, d]\)-code over a finite field \(\mathbb{F}\). Then its minimum distance is given by

\[
d = \min_{\bar{c} \in \mathcal{C} \setminus \{\bar{0}\}} w(\bar{c}).
\]

Proof. Recall that the minimum distance of the code \(\mathcal{C}\) is

\[
d = \min_{\bar{c}_1, \bar{c}_2 \in \mathcal{C}, \bar{c}_1 \neq \bar{c}_2} d(\bar{c}_1, \bar{c}_2).
\]

From the definition of the Hamming weight

\[
d(\bar{c}_1, \bar{c}_2) = w(\bar{c}_1 - \bar{c}_2).
\]

Since \(\mathcal{C}\) is linear, if \(\bar{c}_1, \bar{c}_2 \in \mathcal{C}, \bar{c}_1 \neq \bar{c}_2\), then \(\bar{c}_1 - \bar{c}_2 \in \mathcal{C} \setminus \{\bar{0}\}\). And, vice verse, if \(\bar{c} \in \mathcal{C} \setminus \{\bar{0}\}\), then \(\bar{c} - \bar{0} \in \mathcal{C}, \bar{c} \neq \bar{0}\).

We obtain that

\[
d = \min_{\bar{c}_1, \bar{c}_2 \in \mathcal{C}, \bar{c}_1 \neq \bar{c}_2} d(\bar{c}_1, \bar{c}_2) = \min_{\bar{c}_1, \bar{c}_2 \in \mathcal{C}, \bar{c}_1 \neq \bar{c}_2} w(\bar{c}_1 - \bar{c}_2) = \min_{\bar{c} \in \mathcal{C} \setminus \{\bar{0}\}} w(\bar{c}).
\]

\(\square\)

Encoding of linear codes

Let \(\mathcal{C}\) be a linear \([n, k, d]\)-code over a finite field \(\mathbb{F}\). Let \(G\) be its generator matrix. We define encoding

\[
\mathcal{E} : \mathbb{F}^k \to \mathcal{C} \text{ such that } \bar{x} \mapsto \bar{x} \cdot G.
\]

Since \(\text{rank}(G) = k\), the mapping \(\mathcal{E}\) is one-to-one.

Example. Take \([3, 2, 3]\)-parity code over \(\mathbb{F}_2\). Let

\[
G = \begin{pmatrix}
1 & 0 & 1 \\
0 & 1 & 1
\end{pmatrix}.
\]

If we take \(\bar{x} = (0 1)\), then

\[
\mathcal{E}(\bar{x}) = \bar{x} \cdot G = (0 1) \cdot \begin{pmatrix}
1 & 0 & 1 \\
0 & 1 & 1
\end{pmatrix} = (0 1 1).
\]

Similarly,

\[
\bar{x} = (1 0) : \quad \mathcal{E}(\bar{x}) = \bar{x} \cdot G = (1 0) \cdot \begin{pmatrix}
1 & 0 & 1 \\
0 & 1 & 1
\end{pmatrix} = (1 0 1),
\]

2
\begin{align*}
\bar{x} &= (11) : \ E(\bar{x}) = \bar{x} \cdot G = (11) \cdot \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} = (110), \\
\bar{x} &= (00) : \ E(\bar{x}) = \bar{x} \cdot G = (00) \cdot \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} = (000).
\end{align*}

**Definition.** Systematic generator matrix over \( \mathbb{F} \) is a generator matrix, which has form

\[ G = \begin{pmatrix} I_k \\ A \end{pmatrix}, \]

where \( A \) is a \( k \times (n-k) \) submatrix over \( \mathbb{F} \), and

\[ I_k = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \]

is the \( k \times k \) identity matrix.

**Remarks.**

- If there exists a systematic generator matrix for some code \( C \), then it is unique.
- \( C \) has a systematic generator matrix if the first \( k \) columns of any generator matrix \( G \) of \( C \) are linearly independent (then it is possible to obtain the systematic generator matrix by performing Gaussian elimination on the rows of \( G \)).
- Otherwise, we can always permute columns such that the first \( k \) columns will be independent.

The encoding using systematic generator matrix takes a form

\[ \bar{x} \mapsto \bar{x} \cdot G = (\bar{x} \vert \bar{x}A), \]

where \( \bar{x} \) is the original information and \( \bar{x}A \) are redundant symbols.

**Parity-check matrix**

Let \( C \) be an \([n,k,d] \) code over a finite field \( \mathbb{F} \). A parity-check matrix of \( C \) is \( r \times n \) matrix \( H \) over \( \mathbb{F} \) such that for every \( \bar{c} \in \mathbb{F}^n \)

\[ \bar{c} \in C \iff H \cdot \bar{c}^T = \bar{0}^T. \]

Here \( \bar{c}^T \) denotes the transpose of the row vector \( \bar{c} \).

**Example.** Consider the binary \([3,1,3] \) repetition code. Then a possible parity-check matrix is

\[ H = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \quad \text{and} \quad r = 2. \]

Take \( \bar{c} = (1\ 1\ 1) \), then

\[ H \cdot \bar{c}^T = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \]
Generally,

\[ \text{rank}(H) = n - \dim(\ker(H)) = n - k. \]

If the matrix \( H \) is a full rank then \( r = n - k \).
Extension Fields

Definition Φ is an extension field of F if it contains all elements of F and the operations of Φ on these elements coincide with operations of F.

Example

We construct the field of residues of polynomials over \( \mathbb{F}_2 \) modulo the irreducible polynomial \( P(x) = x^3 + x + 1 \). The field that we construct will be denoted \( \mathbb{F}_8 \) or \( \mathbb{F}_{2^3} \).

From the fact that \( P(x) = 0 \mod P(x) \) we can establish a simple identity

\[
x^3 + x + 1 = 0 \mod P(x) \\
x^3 = x + 1 \mod P(x) \quad \text{(note that } -(x + 1) = x + 1 \text{ in } \mathbb{F}_2)\]

Clearly the field contains the additive and multiplicative neutral elements 0 and 1, let’s assume that \( \beta \) is an element in \( \mathbb{F}_8 \) such that \( \beta \neq 0 \) and \( \beta \neq 1 \). Let’s find consecutive powers of this element:

\[
\begin{align*}
\beta^2 &= \beta^2 \mod P(x) \\
\beta^3 &= \beta + 1 \mod P(x) \quad \text{(by the above identity)} \\
\beta^4 &= \beta \beta^3 = \beta(\beta + 1) = \\
&= \beta^2 + \beta \mod P(x) \\
\beta^5 &= \beta \beta^4 = \beta(\beta^2 + \beta) = \beta^3 + \beta^2 = \\
&= \beta^2 + \beta + 1 \mod P(x) \\
\beta^6 &= (\beta^3)^2 = (\beta + 1)(\beta + 1) = \beta^2 + \beta + \beta + 1 = \\
&= \beta^2 + 1 \mod P(x)
\end{align*}
\]

By continuing this pattern, we will get repeating results (try finding \( \beta^7 \)).

We now have a total of 8 elements:

<table>
<thead>
<tr>
<th>Powers of ( \beta )</th>
<th>Elements of the field</th>
<th>Vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>000</td>
</tr>
<tr>
<td>( \beta^0 )</td>
<td>1</td>
<td>001</td>
</tr>
<tr>
<td>( \beta^1 )</td>
<td>( \beta )</td>
<td>010</td>
</tr>
<tr>
<td>( \beta^2 )</td>
<td>( \beta^2 )</td>
<td>100</td>
</tr>
<tr>
<td>( \beta^3 )</td>
<td>( \beta + 1 )</td>
<td>011</td>
</tr>
<tr>
<td>( \beta^4 )</td>
<td>( \beta^2 + \beta )</td>
<td>110</td>
</tr>
<tr>
<td>( \beta^5 )</td>
<td>( \beta^2 + \beta + 1 )</td>
<td>111</td>
</tr>
<tr>
<td>( \beta^6 )</td>
<td>( \beta^2 + 1 )</td>
<td>101</td>
</tr>
</tbody>
</table>

Note We chose \( P(x) = x^3 + x + 1 \), but we could have chosen \( P(x) = x^3 + x^2 + 1 \) (also irreducible) in which case we would have obtained a different table, but the fields would be isomorphic.
Note The field $\mathbb{F}_8$ is different from the field $\mathbb{Z}_8$ (integers modulo 8).
The extension field $\mathbb{F}_8$ is a vector space over $\mathbb{F}$.

$$k_2\beta^2 + k_1\beta + k_0 \sim (k_2, k_1, k_0) \in (\mathbb{F}_2)^3$$

The set $\{\beta^2, \beta, 1\}$ is one possible basis for that vector space and the third column in the table above represents the coordinates of the elements with respect to this basis.

Example additions and multiplications in $\mathbb{F}_8$

$$\beta^2 + \beta^4 = \beta^2 + (\beta^2 + \beta) = \beta$$
$$\beta^3 + \beta^5 = \beta + 1 + \beta^2 + \beta + 1 = \beta^2$$
$$\beta^2\beta^4 = \beta^6 = \beta^2 + 1$$
$$\beta^4\beta^6 = \beta^{10} = \beta^7\beta^3 = 1 \cdot \beta^3 = \beta + 1$$

Polynomial roots

Definition Let $\mathbb{F}$ be a field and $\Phi$ its extension field. Element $\beta \in \Phi$ is a root of a polynomial $a(x) \in \mathbb{F}[x]$ if $a(\beta) = 0$ in the field $\Phi$.

Taking the polynomial $P(x) = x^3 + x + 1$ and the extension field from the previous section, we can show that $\beta, \beta^2$ and $\beta^4$ are its roots:

$$P(\beta) = \beta^3 + \beta + 1 = \beta + 1 + \beta + 1 = 0 \mod P(x)$$
$$P(\beta^2) = \beta^6 + \beta^2 + 1 = \beta^2 + 1 + \beta^2 + 1 = 0 \mod P(x)$$
$$P(\beta^4) = \beta^{12} + \beta^4 + 1 = \beta^5\beta^7 + \beta^2 + \beta + 1 = \beta^2 + \beta + 1 + \beta^2 + \beta + 1 = 0 \mod P(x)$$

Theorem Let $a(x) \in \mathbb{F}[x]$ and $\beta \in \Phi$. Then $a(\beta) = 0$ if and only if $(x - \beta) | a(x)$.

Proof There exist $d(x), r \in \mathbb{F}[x]$ such that

$$a(x) = d(x) \cdot (x - \beta) + r \quad \deg(r) < 1 \Rightarrow r \in \mathbb{F}$$

Assume $a(\beta) = 0$ then $d(\beta) \cdot (\beta - \beta) + r = 0 \Rightarrow r = 0 \Rightarrow a(x) = d(x) \cdot (x - \beta) \Rightarrow (x - \beta) | a(x)$.

Assume $(x - \beta) | a(x)$ then $a(x) = d(x) \cdot (x - \beta)$ and $a(\beta) = d(\beta) \cdot (\beta - \beta) = 0$. \hfill $\square$

By the theorem above, we can write the polynomial $P(x) = x^3 + x + 1$ as a product

$$P(x) = (x - \beta)(x - \beta^2)(x - \beta^4)d(x).$$

Note that $d(x) = 1$ because the degree and the leading coefficient of both sides have to be equal.

Note Despite the fact that the polynomial $P(x)$ is irreducible over $\mathbb{F}_2$, it is reducible over $\mathbb{F}_{23}$.

Theorem Let $\mathbb{F}$ be a finite field. Then

$$\prod_{\alpha \in \mathbb{F}} (x - \alpha) = x^{||\mathbb{F}||} - x.$$

Proof From question 3 in homework 1 we know that $\alpha^{||\mathbb{F}||} = \alpha$ for any $\alpha \in \mathbb{F}$, therefore every $\alpha$ is a root of $x^{||\mathbb{F}||} - x$. Then by previous theorem $(x - \alpha) | x^{||\mathbb{F}||} - x$ for any $\alpha$. Since the polynomials on both sides of the equation are monic and have the same degree, they must be equal. \hfill $\square$
**Definition** The **multiplicity** of root $\beta$ in $a(x)$ is the largest $m \in \mathbb{N}$ such that $(x - \beta)^m | a(x)$.

**Theorem** A polynomial of degree $n \geq 0$ over a field $\mathbb{F}$ has at most $n$ roots (counting multiplicities) in every extension field of $\mathbb{F}$.

**Definition** An element, whose powers generate all non-zero elements of a field, is a **primitive element** in that field.

**Theorem** Every finite field contains a primitive element.
Generator matrix and parity-check matrix

We continue with the discussion on generator matrices and parity-check matrices of linear codes.

Both matrices, a generator matrix $G$ and a parity-check matrix $H$, define a corresponding code in a unique way. For the generator matrix $G$, the code is a linear span of its rows. For the parity-check matrix $H$, the code is its null-space (kernel). On the other hand, usually linear code has many generator matrices and many parity-check matrices.

Let $C$ be a linear code over a finite field $\mathbb{F}$. A parity-check matrix $H$ of $C$ can be computed from its generator matrix $G$ by using the following identity:

$$ H \cdot G^T = 0^T, \quad \text{or} \quad G \cdot H^T = 0^T. \quad (1) $$

Similarly, a generator matrix $G$ of $C$ can be computed from $H$ by using the same identity.

**Example.** Let $C$ be a linear code over $\mathbb{F}$. Assume that the $k \times n$ generator matrix of $C$, $G$, has the form

$$ G = ( I \mid A ), $$

where $I$ is a $k \times k$ identity matrix and $A$ is a $k \times (n - k)$ matrix with entries in $\mathbb{F}$. Then, the corresponding $(n - k) \times n$ parity-check matrix is

$$ H = ( -A^T \mid I ). $$

To verify (1), we compute

$$ G \cdot H^T = ( I \mid A ) \left( \frac{-A}{I} \right)^T = -A + A = 0. $$

**Example.** An $[n, n - 1, 2]$ parity code over $\mathbb{F}_2$. The words of the code are all even-weight vectors of length $n$. An $1 \times n$ parity-check matrix is given by

$$ H = ( 1 \ 1 \ 1 \ldots 1 ). $$

**Example.** An $[n, 1, n]$ repetition code over $\mathbb{F}_2$. An $(n - 1) \times n$ parity-check matrix is given by

$$ H = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 1 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 1 & \cdots & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{pmatrix}. $$
Remark. We see from the above examples that the generator matrix of the parity code is a parity-check matrix of the repetition code. And vice versa: the generator matrix of the repetition code is a parity-check matrix of the parity code. This motivates the following definition.

Definition. Let $G$ be a generator matrix of a linear code $C$ of dimension $k$ over $\mathbb{F}$, and let $H$ be an $(n - k) \times n$ parity-check matrix of $C$. The code, whose generator matrix is $H$, is called the dual code of $C$ and is denoted by $C^\perp$. The code $C$ itself is called a primal code. The codes $C$ and $C^\perp$ are also called a dual pair.

Assume that $H$ is a full-rank matrix. The following table summarizes the connections between the primal and the dual codes, and their corresponding generator and parity-check matrices.

<table>
<thead>
<tr>
<th>Code</th>
<th>Generator matrix</th>
<th>Parity-check matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C$</td>
<td>$G$</td>
<td>$H$ (full rank)</td>
</tr>
<tr>
<td>$C^T$</td>
<td>$H$ (full rank)</td>
<td>$G$</td>
</tr>
</tbody>
</table>

Note. For any code $C$, $(C^\perp)^\perp = C$.

Corollary. The $[n, 1, n]$ repetition code over $\mathbb{F}_2$ and the $[n, n - 1, 2]$ parity code over $\mathbb{F}_2$ are a dual pair.

Example. The $[7, 4, 3]$ Hamming code over $\mathbb{F}_2$ is defined by the following parity-check matrix:

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

The matrix $H$ is $3 \times 7$ matrix. Its columns are all non-zero vectors of length 3.

A corresponding generator matrix is:

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

Indeed, $H \cdot G^T = 0^T$. Moreover, $\dim(\text{Ker}(H)) = 7 - \text{rank}(H) = 7 - 3 = 4$. It can be shown that the matrix $G$ is a full-rank matrix, and so its rank is 4. Therefore, a linear span of the rows of $G$ is exactly the null-space of $H$.

The minimum distance of the Hamming code can be verified by checking Hamming weights of all codewords.

Remark. How do we find $G$ if we know $H$?

1. Solve the system of linear equations $H \cdot x^T = 0^T$.
2. Write the basis vectors of the solution space as the rows of $G$. 


Minimum distance of the code given by its parity-check matrix

**Theorem.** Let $H$ be a parity check matrix of a linear code $C \neq \{0\}$ over $\mathbb{F}$. The minimum distance of $C$ is the largest $d \in \mathbb{N}$ such that any set of $d - 1$ columns in $H$ is linearly independent.

**Proof.**

1. Write

$$H = \begin{pmatrix} \bar{h}_1 & \bar{h}_2 & \cdots & \bar{h}_n \end{pmatrix},$$

where $\bar{h}_i^T$ is the $i$-th column in $H$. Let $\bar{c} = (c_1, \ldots, c_n)$ be a codeword of the Hamming weight $t > 0$. Let $J$ be a set of non-zero coordinates in $\bar{c}$.

$$H \cdot \bar{c}^T = \bar{0}^T \iff \sum_{i \in J} c_i \bar{h}_i^T = \bar{0}.$$

Therefore, $t$ columns indexed by $J$ are linearly dependent. We obtain that there exist $d$ columns in $H$ that are linearly dependent.

2. Take any set of linearly dependent columns in $H$. Denote it by $J$. Then there exist coefficients $c_i \in \mathbb{F}$, $i \in J$, such that

$$\sum_{i \in J} c_i \bar{h}_i^T = \bar{0}^T. \tag{2}$$

If the number of columns represented by $J$ is $t$, then (2) contains at most $t$ non-zero vectors. The coefficients in (2) together with additional zeros form a vector $\bar{c}$ of weight $\leq t$. This is not possible with $t \leq d - 1$ linearly dependent columns.

We obtain that any $d - 1$ columns are linearly independent. Therefore, the smallest possible set of linearly dependent columns is of size $d$.

$\square$

**Example.** Let $m > 1$. The $[2^m - 1, 2^m - m - 1, 3]$ Hamming code over $\mathbb{F}_2$ is defined by an $m \times (2^m - 1)$ parity-check matrix, whose columns range over all nonzero vectors in $(\mathbb{F}_2)^m$.

$$H = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & \cdots & 1 \\
0 & 0 & 0 & 0 & 0 & \cdots & 1 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 1 & 1 & \cdots & 1 \\
0 & 1 & 1 & 0 & 0 & \cdots & 1 \\
1 & 0 & 1 & 0 & 1 & \cdots & 1 \end{pmatrix}$$

Any two columns of $H$ are different, and therefore are linearly independent. The first three columns of $H$ are, on the other hand, dependent. Therefore, $d = 3$. 

3
Extended Hamming code

Recall a $3 \times 7$ parity-check matrix of the binary $[7, 4, 3]$ Hamming code:

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

We construct a parity-check matrix of a new code by adding an all-zero column to $H$, and then by adding an all-one row to the obtained matrix. The resulting $4 \times 8$ matrix has a form:

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

A code $\hat{C}$ defined by the matrix $\hat{H}$ is called an extended binary Hamming code.

What are the parameters of the new code?

- The length of the new code is obviously 8.
- Observe that the first four columns in $\hat{H}$ constitute a sub-matrix which has a form

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

and after reordering of the columns it becomes an upper triangular matrix

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

Therefore, the rank of $\hat{H}$ is full. We obtain that $n - k = 4$, and $k = 4$.

- What is the smallest weight of a nonzero vector in $\hat{C}$?

Take a nonzero codeword $\bar{c} = (c_1, c_2, \cdots, c_8) \in \hat{C}$. Its sub-word $(c_2, c_3, \cdots, c_8)$ is not equal to $\bar{0}$ (otherwise, we also have that $c_1 = 0$ from the first row of $\hat{H}$).

From the last three rows of $\hat{H}$ we see that $(c_2, c_3, \cdots, c_8)$ is a codeword of a binary $[7, 4, 3]$ Hamming code. The Hamming weight of this sub-word is at least 3. If it is exactly 3, then the first row of $\hat{H}$ ensures that $c_1 = 1$. Therefore, in any case the weight of $\bar{c}$ is at least 4.
Remark. The matrix $\hat{H}$ is also a generator matrix of $\hat{C}$. To see that, we can multiply $\hat{H}$ by any transposed row of $\hat{H}$ and verify that indeed we obtain the zero vector. Therefore, the code $\hat{C}$ is dual to itself or self-dual.

Concatenated codes

Now, we present another to construct new codes from other codes. For this construction, we will need the following three ingredients.

- Inner code $C_{\text{in}}$ is a linear $[n, k, d]$ code over $\mathbb{F}_q$.
- Outer code $C_{\text{out}}$ is a linear $[N, K, D]$ code over $\mathbb{F}_{q^k}$ (extension field of $\mathbb{F}_q$ of extension degree $k$).
- Linear mapping (bijection) $E : \mathbb{F}_{q^k} \rightarrow C_{\text{in}}$. This mapping associates symbols in $\mathbb{F}_{q^k}$ with codewords in $C_{\text{in}}$.

**Definition 1.** A concatenated code $C_{\text{cont}}$ over $\mathbb{F}_q$ is defined as follows:

$$C_{\text{cont}} = \{ (E(c_1) \mid E(c_2) \mid \cdots \mid E(c_N)) \in (\mathbb{F}_q)^{N \cdot n} : (c_1, c_2, \cdots, c_N) \in C_{\text{out}} \}.$$ 

**Example.** Take $\bar{c} = (c_1, c_2, \ldots, c_N) \in C_{\text{out}}$. 

\[
\begin{array}{cccc}
\vdots & \vdots & \vdots & \vdots \\
E(c_1) \in C_{\text{in}} & E(c_2) \in C_{\text{in}} & E(c_3) \in C_{\text{in}} & \ldots & E(c_N) \in C_{\text{in}}
\end{array}
\]

\[
\begin{pmatrix}
\vdots \\
(c_1 \mid c_2 \mid c_3 \mid \cdots \mid c_N)
\end{pmatrix}
\]

**Example.** Let $C_{\text{in}}$ be a parity code of length 3 over $\mathbb{F}_2$ with a generator matrix

$$G_{\text{in}} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$ 

Let $C_{\text{out}}$ be the code of length 3 over $\mathbb{F}_{2^2}$ defined by the generator matrix

$$G_{\text{out}} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & \beta \end{pmatrix},$$

where $\mathbb{F}_{2^2} = \{0, 1, \beta, \beta+1\}$.

Take the mapping $E$ to be as follows:

\[
\begin{cases}
E(1) = (011) \\
E(\beta) = (101)
\end{cases}
\]

Due to linearity of $E$ it follows that $E(0) = (000)$ and $E(\beta+1) = (110)$.

- Take $(0 \ 1 \ \beta) \in C_{\text{out}}$. Then $(0 \ 0 \ 0 \mid 0 \ 1 \ 1 \mid 1 \ 0 \ 1) \in C_{\text{cont}}$.
- Take $(1 \ 1 \ \beta+1) \in C_{\text{out}}$. Then $(0 \ 1 \ 1 \mid 0 \ 1 \ 1 \mid 1 \ 1 \ 0) \in C_{\text{cont}}$. 

2
Analysis of parameters of $C_{cont}$

Next, we analyze the parameters $[N_{cont}, K_{cont}, D_{cont}]$ of the code of $C_{cont}$.

- It is easy to see that $N_{cont} = N \cdot n$.
- What is the number of the codewords in $C_{cont}$? Since the mapping $E(\cdot)$ is a bijection between $\mathbb{F}_{q^k}$ and $C_{in}$, we have that

$$|C_{cont}| = |C_{out}| = (q^k)^K = q^{k \cdot K},$$

and so

$$K_{out} = \log_q q^{k \cdot K} = k \cdot K.$$

- Next, we estimate the minimum distance of $C_{cont}$. Take $\bar{c}' = (E(c_1) \mid E(c_2) \mid \cdots \mid E(c_N)) \in C_{cont}$, $\bar{c}' \neq \bar{0}$, where $(c_1, c_2, \cdots, c_N) \in C_{out}$. The vector $(c_1, c_2, \cdots, c_N)$ contains at least $D$ nonzero coordinates. Each of $c_1, c_2, \cdots, c_N$ (those that are not zeros) are mapped onto nonzero codewords of length $n$ in $C_{in}$. The nonzero images in $\{E(c_1), E(c_2), \cdots, E(c_N)\}$ all have Hamming weights at least $d$. We obtain that $w(\bar{c}') \geq D \cdot d$, and so $D_{cont} \geq D \cdot d$.

Decoding of linear codes

Let $C$ be a linear $[n, k, d]$ code over $\mathbb{F}_q$, and let $H$ be its $(n - k) \times n$ parity-check matrix.

The nearest-neighbor decoding:

Given $\bar{y} \in (\mathbb{F}_q)^n$, find a codeword $\bar{c} \in C$, which minimizes $d(\bar{y}, \bar{c})$.

Alternative (equivalent) formulation:

Given a received word $\bar{y} \in (\mathbb{F}_q)^n$, find a word $\bar{e} \in (\mathbb{F}_q)^n$ that has a minimal possible Hamming weight, and such that $\bar{y} - \bar{e} \in C$.

Definition. We define a syndrome $\bar{s}$ of the word $\bar{y} \in (\mathbb{F}_q)^n$, a column vector of length $n - k$ over $(\mathbb{F}_q)$, as

$$\bar{s} = H \cdot \bar{y}^T.$$

Recall that

$$\bar{c} \in C \iff H \cdot \bar{c}^T = \bar{0}^T.$$

Therefore, the codewords are the vectors in $(\mathbb{F}_q)^n$, whose syndrome is exactly $\bar{0}^T$.

More generally, if $\bar{y}_1, \bar{y}_2 \in (\mathbb{F}_q)^n$, such that $\bar{y}_1 - \bar{y}_2 \in C$, then

$$\bar{0} = H \cdot (\bar{y}_1^T - \bar{y}_2^T) = H \cdot \bar{y}_1^T - H \cdot \bar{y}_2^T,$$

and so $H \cdot \bar{y}_1^T = H \cdot \bar{y}_2^T$.

Therefore, $\bar{y}_1$ and $\bar{y}_2$ are in the same coset of $C$ in $(\mathbb{F}_q)^n$. 

3
**Syndrome decoding**

Suppose that $C$ is an $[n,k,d]$ linear code with a parity-check matrix $H$. Let $\bar{c} \in C$ be transmitted, and $\bar{y} \in (\mathbb{F}_q)^n$ be received. Then, the *syndrome decoding* method works as follows:

1. Find the syndrome $\bar{s} = H \cdot \bar{y}^T$.
2. Find the coset leader (the minimum weight solution) $\bar{e} \in (\mathbb{F}_q)^n$, such that $\bar{s} = H \cdot \bar{e}^T$.

For general linear codes, the second step is NP-hard. However, for some codes this step can be done efficiently.
Bounds on the parameters of the code

Next, we present several simple bounds on the parameters of codes.

Singleton bound

**Theorem.** Let $C$ be an $(n, M, d)$ code over a finite field $\mathbb{F}_q$. Then $M \leq q^{n-d+1}$.

**Proof.** Assume by contrary that $M > q^{n-d+1}$. Then there are two codewords $\bar{u} \in C$ and $\bar{v} \in C$ that agree on their first $n - d + 1$ coordinates. Thus, the Hamming distance between those two codewords is at most $d - 1$. This is a contradiction, since we assumed that the minimum Hamming distance of $C$ is $d$.

Singleton bound for a linear $[n, k, d]$ code over $\mathbb{F}_q$ can be restated as follows:

$$q^k \leq q^{n-d+1},$$

or

$$k + d - 1 \leq n.$$

Examples of codes that attains the Singleton bound with equality include:

- The whole space $(\mathbb{F}_q)^n$ can be viewed as a linear $[n, n, 1]$ code over $\mathbb{F}_q$.
- The $[n, n-1, 2]$ parity code over $\mathbb{F}_q$.
- The $[n, 1, n]$ repetition code over $\mathbb{F}_q$.
- Reed-Solomon code (will be studied later).

Sphere-packing/Hamming bound

We start with the following definition.

**Definition.** A sphere of radius $t > 0$ around a vector $\bar{v} \in (\mathbb{F}_q)^n$ is defined as

$$S_{t,n}(\bar{v}) = \{ \bar{x} \in (\mathbb{F}_q)^n : d(\bar{x}, \bar{v}) \leq t \}.$$
That sphere contains all vectors in \((\mathbb{F}_q)^n\), which are located at the Hamming distance at most \(t\) from \(\bar{v}\).

For any \(\bar{v} \in (\mathbb{F}_q)^n\), the size of a sphere of radius \(t > 0\) is given by the following expression:

\[
S_{t,n} = |S_{t,n}(\bar{v})| = \sum_{i=0}^{t} \binom{n}{i} (q - 1)^i.
\]

The size of the sphere does not depend on the choice of the vector \(\bar{v}\).

Let \(\mathcal{C}\) be an \((n, M, d)\) code over \((\mathbb{F}_q)^n\). Consider a collection of spheres of radius \(\lfloor \frac{d-1}{2} \rfloor\) around the codewords of \(\mathcal{C}\).

These spheres have to be disjoint. Otherwise, assume that the spheres around codewords \(\bar{c}_i \in \mathcal{C}\) and \(\bar{c}_j \in \mathcal{C}\), \(\bar{c}_i \neq \bar{c}_j\), have a nonempty intersection. Take a vector \(\bar{x}\) in this intersection. We obtain that \(d(\bar{x}, \bar{c}_i) \leq \lfloor \frac{d-1}{2} \rfloor\) and \(d(\bar{x}, \bar{c}_j) \leq \lfloor \frac{d-1}{2} \rfloor\). The triangle inequality yields that \(d(\bar{c}_i, \bar{c}_j) \leq d - 1\), in contradiction to the fact that \(d\) is the minimum distance of \(\mathcal{C}\). We conclude that the spheres are disjoint indeed.

The picture below illustrates this situation.

\[\text{Since all spheres have to be disjoint, and all together are contained in } (\mathbb{F}_q)^n, \text{ we obtain the following inequality, which is known as Hamming bound or sphere-packing bound.}\]

\[
M \cdot S_{\lfloor \frac{d-1}{2} \rfloor,n} \leq q^n,
\]
or, equivalently,
\[ M \cdot \sum_{i=0}^{\lfloor \frac{d-1}{2} \rfloor} \binom{n}{i} (q-1)^i \leq q^n. \]

If \( C \) is a linear \([n, k, d]\) code, then the Hamming bounds becomes:
\[ \sum_{i=0}^{\lfloor \frac{d-1}{2} \rfloor} \binom{n}{i} (q-1)^i \leq q^{n-k}. \]

The code that attains this bound with equality is called perfect.

**Example.** Let \( C \) be a binary \([n, 1, n]\) repetition code, and \( n \) be an odd integer.
\[ S_{(n-1)/2,n} = \frac{n-1}{2} \sum_{i=0}^{n-1} \binom{n}{i} = \sum_{j=\frac{n-1}{2}+1}^{n} \binom{n}{j}, \]
where the last transition is obtained by rearrangement of binomial coefficients.

Therefore,
\[ S_{(n-1)/2,n} = \frac{1}{2} \sum_{i=0}^{n-1} \binom{n}{i} = 2^{n-1}, \]
or
\[ 2 \cdot S_{(n-1)/2,n} = 2^n. \]

Therefore, the code \( C \) is perfect.

**Example.** Let \( C \) be a binary \([n, n-m, 3]\) Hamming code, where \( m > 1 \) and \( n = 2^m - 1 \).
\[ S_{1,n} = 1 + n = 2^m. \]

We obtain that
\[ 2^{n-m} \cdot S_{1,n} = 2^n, \]
and therefore the code \( C \) is perfect.

**Example.** Other perfect codes include:

- The \( q \)-ary Hamming code.
- The \([23, 12, 7]\) Golay code over \( \mathbb{F}_2 \) (will not be studied in the course).
- The \([11, 6, 5]\) Golay code over \( \mathbb{F}_3 \) (will not be studied in the course).
Gilbert-Varshamov bound

**Theorem.** Let $\mathbb{F}_q$ be a finite field and let $n, k, d$ be such that
\[
S_{d-2,n-1} < q^{n-k}.
\]
(1)

Then there exists a linear $[n, k, \geq d]$ code over $\mathbb{F}_q$.

**Proof.** We construct an $(n-k) \times n$ parity-check matrix $H$ for this code, such that any $d-1$ columns are linearly independent. We start with $(n-k) \times (n-k)$ identity matrix. Assume that we have selected $\ell-1$ columns, denoted as $\bar{h}_1, \bar{h}_2, \ldots, \bar{h}_{\ell-1}$. A vector in $\mathbb{F}_q^{n-k}$ can not be selected as an $\ell$-th column if and only if it can be expressed as a linear combination of any $d-2$ existing columns from $\{\bar{h}_1, \bar{h}_2, \ldots, \bar{h}_{\ell-1}\}$. The number of such linear combinations is:
\[
\sum_{i=0}^{d-2} \binom{\ell-1}{i} (q-1)^i,
\]
which is equal to $S_{d-2,\ell-1}$.

In order to be able to select an $\ell$-th column, it is sufficient to require that $S_{d-2,\ell-1} < q^{n-k}$ (i.e. there exists a suitable vector).

From (1), we have that $S_{d-2,n-1} < q^{n-k}$. However, in that case, for any $\ell \leq n$ we have $S_{d-2,\ell-1} \leq S_{d-2,n-1} < q^{n-k}$, which means that we are able to add a column to $H$. 

\[\square\]
Asymptotic versions of the bounds

Let $C$ be a linear $[n, k, d]$ code over a finite field $\mathbb{F}$ with $q$ elements. In the previous lecture, we derived several bounds on the parameters of $C$.

**Singleton bound:**

$$k + d - 1 \leq n.$$  

**Sphere-packing (Hamming) bound:**

$$S_{\left\lfloor \frac{d-1}{2} \right\rfloor, n} \leq q^{n-k},$$

where

$$S_{t,n} \triangleq \sum_{i=0}^{t} \binom{n}{i} \cdot (q-1)^i.$$  

**Gilbert-Varshamov bound:** Let $n, k$ and $d$ be positive integers, such that

$$S_{d-2,n-1} < q^{n-k}. \quad (1)$$

Then, there exists a linear $[n, k, \geq d]$ code over $\mathbb{F}$.

The first two bounds are upper bounds on the parameters of the code (every code satisfies them). The third bound is a lower bound (i.e., there exists a code that satisfies that bound).

Next, we turn to the asymptotic versions of these three bounds. We consider a regime where $n \to \infty$. We use the following notations.

- The rate of the code: $R = \frac{k}{n}$.
- The relative minimum distance of the code: $\delta = \frac{d}{n}$.

The asymptotic version of these bounds are presented below.

**Singleton bound:**

$$R \leq 1 - \delta + o(1).$$

**Sphere-packing (Hamming) bound:**

$$R \leq 1 - H_q(\delta/2) + o(1)$$

where

$$H_q(x) = -x \log_q x - (1-x) \log_q (1-x) + x \log_q (q-1)$$

is the $q$-ary entropy function.
Gilbert-Varshamov bound: If $R = 1 - H_q(\delta)$, then there exists a code of sufficiently large length $n$ with rate $\geq R$ and relative minimum distance $\geq \delta$.

Reed-Solomon code

Definition. A code that achieves the Singleton bound, is called maximum distance separable (MDS).

Let $\mathbb{F}_q$ be a finite field, and $\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_n \in \mathbb{F}_q$ be different nonzero elements (called code locators), $q \geq n+1$. An $[n, k, d]$ Reed-Solomon code over $\mathbb{F}_q$ is defined by its $(n-k) \times n$ parity-check matrix as follows:

$$H_{RS} = \begin{pmatrix}
1 & 1 & 1 & \ldots & 1 \\
\alpha_1 & \alpha_2 & \alpha_3 & \ldots & \alpha_n \\
\alpha_1^2 & \alpha_2^2 & \alpha_3^2 & \ldots & \alpha_n^2 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\alpha_1^{n-k-1} & \alpha_2^{n-k-1} & \alpha_3^{n-k-1} & \ldots & \alpha_n^{n-k-1}
\end{pmatrix}.$$ 

Theorem. An $[n, k, d]$ Reed-Solomon code over $\mathbb{F}_q$ is an MDS code.

In order to prove this theorem, we first formulate and prove the following lemma.

Lemma. If an $(n-k) \times (n-k)$ matrix $B$ over a field $\mathbb{F}_q$ is of the Vandermonde form:

$$B = \begin{pmatrix}
1 & 1 & 1 & \ldots & 1 \\
\beta_1 & \beta_2 & \beta_3 & \ldots & \beta_{n-k} \\
\beta_1^2 & \beta_2^2 & \beta_3^2 & \ldots & \beta_{n-k}^2 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\beta_1^{n-k-1} & \beta_2^{n-k-1} & \beta_3^{n-k-1} & \ldots & \beta_{n-k}^{n-k-1}
\end{pmatrix},$$

where $\beta_1, \beta_2, \ldots, \beta_n \in \mathbb{F}_q$ are all distinct nonzero elements in $\mathbb{F}_q$, then

$$\det(B) = \prod_{1 \leq i < j \leq n} (\beta_j - \beta_i).$$

Proof. The proof is by induction on $n-k$.

Basis. If $n-k = 1$, then

$$B = \begin{pmatrix} 1 \end{pmatrix},$$

and the claim is trivially true.

If $n-k = 2$, then

$$B = \begin{pmatrix} 1 & 1 \\ \beta_1 & \beta_2 \end{pmatrix},$$

and $\det(B) = \beta_2 - \beta_1$. Thus, the claim is true also in this case.

Step.
1. We apply the following column operations to $B$ (the columns are numbered $1, 2, \ldots, n-k$):

$$c_2 \leftarrow c_2 - c_1;$$
$$c_3 \leftarrow c_3 - c_1;$$
$$\cdots$$
$$c_{n-k} \leftarrow c_{n-k} - c_1.$$

and obtain the following matrix with the same determinant

$$B' = \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 \\
\beta_1 & \beta_2 - \beta_1 & \beta_3 - \beta_1 & \cdots & \beta_{n-k} - \beta_1 \\
\beta_2 & \beta_2^2 - \beta_1^2 & \beta_3^2 - \beta_1^2 & \cdots & \beta_{n-k}^2 - \beta_1^2 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\beta_{n-k-1} & \beta_{n-k-1}^2 - \beta_{n-k-1} & \beta_{n-k-1}^2 - \beta_{n-k-1} & \cdots & \beta_{n-k}^2 - \beta_1^2
\end{pmatrix}. $$

2. Next, we apply the following row operations to $B'$ (the rows are numbered $1, 2, \ldots, n-k$):

$$r_2 \leftarrow r_2 - \beta_1 \cdot r_1;$$
$$r_3 \leftarrow r_3 - \beta_1 \cdot r_2;$$
$$\cdots$$
$$r_{n-k} \leftarrow r_{n-k} - \beta_1 \cdot r_{n-k-1}.$$

It is straightforward to verify that we obtain the following matrix\(^1\)

$$B'' = \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 \\
0 & \beta_2 - \beta_1 & \beta_3 - \beta_1 & \cdots & \beta_{n-k} - \beta_1 \\
0 & \beta_2(\beta_2 - \beta_1) & \beta_3(\beta_3 - \beta_1) & \cdots & \beta_{n-k}(\beta_{n-k} - \beta_1) \\
0 & \beta_2^2(\beta_2 - \beta_1) & \beta_3^2(\beta_3 - \beta_1) & \cdots & \beta_{n-k}^2(\beta_{n-k} - \beta_1) \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \beta_{n-k-2}^2(\beta_2 - \beta_1) & \beta_{n-k-2}^2(\beta_3 - \beta_1) & \cdots & \beta_{n-k}^2(\beta_{n-k} - \beta_1)
\end{pmatrix}. $$

3. To this end,

$$\det(B) = \det(B'') = \prod_{j=2}^{n-k} (\beta_j - \beta_1) \cdot \det \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 \\
0 & 1 & 1 & \cdots & 1 \\
0 & \beta_2 & \beta_3 & \cdots & \beta_{n-k} \\
0 & \beta_2^2 & \beta_3^2 & \cdots & \beta_{n-k}^2 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \beta_{n-k-2}^2 & \beta_{n-k-2}^2 & \cdots & \beta_{n-k}^2
\end{pmatrix}$$

$$= \prod_{j=2}^{n-k} (\beta_j - \beta_1) \cdot \prod_{2 \leq i < j \leq n} (\beta_j - \beta_i)$$

$$= \prod_{1 \leq i < j \leq n} (\beta_j - \beta_i),$$

\(^1\)For example, the entry in the third row and the second column of $B''$ is $(\beta_2^2 - \beta_1^2) - \beta_1(\beta_2 - \beta_1) = \beta_2^2 - \beta_1^2 - \beta_1\beta_2 + \beta_1^2 = \beta_2(\beta_2 - \beta_1)$. Other entries in $B''$ are computed in the similar manner.
where the penultimate transition is due to the induction assumption. This completes the proof of the lemma.

\[ \square \]

*Proof of the theorem.* First, observe that every \((n-k) \times (n-k)\) submatrix of \(H_{RS}\) has Vandermonde form. The length of \(\mathcal{C}\) is \(n\).

1. By using the lemma, any \((n-k) \times (n-k)\) submatrix of \(H_{RS}\) is a full-rank, and therefore the rows of \(H_{RS}\) are linearly independent. Then, the dimension of \(\mathcal{C}\) is \(n - (n - k) = k\).

2. By using the theorem from Lecture 6, the minimum distance \(d\) is the largest number such that any \(d-1\) columns in \(H_{RS}\) are linearly independent. In this case, \(d - 1 = n - k\), and so \(d = n - k + 1\).

\[ \square \]
In this lecture, we will discuss Generalized Reed-Solomon codes. We will see the connection to polynomials over corresponding fields, and some interesting mathematical properties that the Generalized Reed-Solomon codes possess. These properties will be used in the decoding algorithms.

**Definition**

A Generalized Reed-Solomon (GRS) \([n, k, d]\) code over a finite field \(\mathbb{F}\) is a code whose generator matrix is of the form

\[
G_{GRS} = \begin{pmatrix}
    u_1 & u_2 & \ldots & u_n \\
    u_1\alpha_1 & u_2\alpha_2 & \ldots & u_n\alpha_n \\
    u_1\alpha_1^2 & u_2\alpha_2^2 & \ldots & u_n\alpha_n^2 \\
    \vdots & \vdots & \ddots & \vdots \\
    u_1\alpha_1^{k-1} & u_2\alpha_2^{k-1} & \ldots & u_n\alpha_n^{k-1}
\end{pmatrix},
\]

where \(u_1, u_2, \ldots, u_n \in \mathbb{F}\) are nonzero field elements, and \(\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}\) are pairwise distinct nonzero field elements (called *error locators*).

The parity-check matrix \(H_{GRS}\) has a very similar form

\[
H_{GRS} = \begin{pmatrix}
    v_1 & v_2 & \ldots & v_n \\
    v_1\alpha_1 & v_2\alpha_2 & \ldots & v_n\alpha_n \\
    v_1\alpha_1^2 & v_2\alpha_2^2 & \ldots & v_n\alpha_n^2 \\
    \vdots & \vdots & \ddots & \vdots \\
    v_1\alpha_1^{n-k+1} & v_2\alpha_2^{n-k+1} & \ldots & v_n\alpha_n^{n-k+1}
\end{pmatrix},
\]

where \(\alpha_1, \alpha_2, \ldots, \alpha_n\) are the same as before, and \(v_1, v_2, \ldots, v_n \in \mathbb{F}\) are nonzero field elements. Yet there are some differences between \(H_{GRS}\) and \(G_{GRS}\):

- **dimension**: there are \(n - k\) rows in \(H_{GRS}\) compared with \(k\) rows in \(G_{GRS}\);
- **multipliers**: the column multipliers \(v_1, v_2, \ldots, v_n\) are generally different from \(u_1, u_2, \ldots, u_n\).

A non-generalized Reed-Solomon code is a special case for \(v_1 = v_2 = \cdots = v_n = 1\). Similarly to non-generalized RS codes, any GRS code is an MDS code (satisfies the Singleton bound \(n = d + k - 1\)).

**Encoding and decoding of GRS codes**

In the first lectures, we have talked about encoding and decoding in general. Let us now consider more specifically the case of GRS codes.
Encoding

Let \( \bar{z} = (z_0, z_1, \ldots, z_{k-1}) \in \mathbb{F}^k \) be the information vector we want to transmit. Generally, the encoding is defined as

\[
E : \bar{z} \to \bar{z} \cdot G_{GRS}.
\]

Let \( \bar{c} = \bar{z} \cdot G_{GRS} \) denote the transmitted vector.

For each vector \( \bar{z} \), we define a polynomial associated with it:

\[
z(x) = z_0 + z_1 x + z_2 x^2 + \ldots + z_{k-1} x^{k-1}.
\]

Then

\[
\bar{c} = \bar{z} \cdot G_{GRS} = \left( \sum_{i=0}^{k-1} u_1 \alpha_1^i z_i, \sum_{i=0}^{k-1} u_2 \alpha_2^i z_i, \ldots, \sum_{i=0}^{k-1} u_n \alpha_n^i z_i \right) = \left( u_1 z(\alpha_1), u_2 z(\alpha_2), \ldots, u_n z(\alpha_n) \right).
\]

The encoding is achieved by evaluating the polynomial \( z(x) \) at the code locators \( \alpha_1, \ldots, \alpha_n \) and then scaling \( z(\alpha_j) \) by \( u_j \).

In general, \( z(x) \) is a polynomial of degree at most \( k - 1 \). This means that it has at most \( k - 1 \) roots in \( \mathbb{F} \). Therefore the codeword \( \bar{c} \) has at most \( k - 1 \) zero entries and at least \( n - (k - 1) \) non-zero entries. We may conclude that the minimum distance of a GRS code is \( d \geq n - (k - 1) = n - k + 1 \). According to the Singleton bound, \( d \leq n - k + 1 \), and hence we get another proof that a GRS code is an MDS code.

Decoding and Error Correction

Let \( \bar{c} \) be the transmitted vector defined as before, and \( \bar{y} \) the received vector that may differ from \( \bar{c} \). The task is to reconstruct \( \bar{z} \) from \( \bar{y} \).

Erasures

If we use erasure channel, then the decoding problem is equivalent to the problem of interpolation of the polynomial \( z(x) \) given its values at some points.
Any polynomial of degree at most $k - 1$ can be reconstructed given at least $k$ distinct points. If the number of known values is at least $k$, then we are able to reconstruct $z(x)$. This means that we can correct $n - k$ erasures. From the Singleton bound, $n - k = d - 1$. Hence we may correct $d - 1$ erasures and thus achieve the maximum theoretical number of erasures (that has been proven earlier).

Errors

Assume that the Hamming distance between $\bar{y}$ and $\bar{c}$ is $t \leq \frac{d - 1}{2}$. Observe that larger value of $t$ may not allow for a unique decoding, as it was shown earlier in the course.

The decoding problem becomes the *noisy interpolation problem*: we want to reconstruct $z(x)$ from its values at different points where some $t$ points have wrong values. The task is to define a polynomial that passes through $n - t$ points, and we need to distinguish correct and incorrect values.
Conventional Reed-Solomon codes

In some literature, these codes are called just Reed-Solomon codes. This is a special case of GRS with some additional restrictions. Let $n$ be the codeword length, $|F| = q$.

- $n \mid q - 1$.
- There exists $\alpha \in F$, an element of multiplicative order $n$ in $F$ ($\alpha^n = 1$).

Let $b$ be some integer. Define the parity-check matrix

$$H_{RS} = \begin{pmatrix}
1 & \alpha^b & \alpha^{2b} & \cdots & \alpha^{(n-1)b} \\
1 & \alpha^{b+1} & \alpha^{2(b+1)} & \cdots & \alpha^{(n-1)(b+1)} \\
1 & \alpha^{b+2} & \alpha^{2(b+2)} & \cdots & \alpha^{(n-1)(b+2)} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & \alpha^{b+d-2} & \alpha^{2(b+d-2)} & \cdots & \alpha^{(n-1)(b+d-2)}
\end{pmatrix},$$

where $b + d - 2 = b + (n - k - 1)$ since the code is an MDS code. The number of rows is $n - k = d - 1$.

We see that the above $H_{RS}$ is a special case of a parity-check matrix of GRS code, $H_{GRS}$, where $\alpha_j = \alpha^{j-1}$, $v_j = \alpha^{b(j-1)}$, for $1 \leq j \leq n$. Denote a Conventional Reed-Solomon code $C_{RS}$.

According to the definition of the parity-check matrix,

$$\bar{c} \in C_{RS} \iff \begin{cases}
\bar{c}(\alpha^b) = 0 \\
\bar{c}(\alpha^{b+1}) = 0 \\
\vdots \\
\bar{c}(\alpha^{b+d-2}) = 0
\end{cases}.$$

In other words, the vector $\bar{c}$ is in $C_{RS}$ if and only if $\alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+d-2}$ are all roots of $c(x)$.

Define a generator polynomial of $C_{RS}$:

$$g(x) = (x - \alpha^b)(x - \alpha^{b+1})\cdots(x - \alpha^{b+d-2}).$$

Then $\bar{c} \in C_{RS}$ if and only if $g(x) \mid c(x)$. The polynomial $c(x)$ can be characterized by the fraction $z(x) = \frac{c(x)}{g(x)}$.

We get an alternative characterization of $C_{RS}$:

$$C_{RS} = \{z(x) \cdot g(x) \mid z(x) \in \mathbb{F}_k[x]\}.$$
All the products \( z(x) \cdot g(x) \) are different due to the counting argument. Since the dimension of \( C_{\text{RS}} \) is \( k \), it contains \( q^k \) codewords that are all represented by distinct polynomials \( c(x) \). The number of polynomials of degree less than \( k \) over a field \( \mathbb{F} \) of size \( q \) is also \( q^k \) (we may choose from \( q \) values of \( \mathbb{F} \) for each of the \( k \) coefficients), so there are \( q^k \) distinct polynomials \( z(x) \). Therefore \( |c(x)| = |z(x)| \). If we had \( z_1(x) \cdot g(x) = z_2(x) \cdot g(x) \), but \( z_1(x) \neq z_2(x) \), we would not get the entire \( C_{\text{RS}} \) from \( \mathbb{F}_k[x] \).

Let us check the degree of the resulting \( c(x) \).

- \( \text{deg}(g(x)) = n - k \) since it has \( n - k \) different nonzero roots: \( \alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+d-2} \).
- \( \text{deg}(z(x)) \leq k - 1 \).
- We obtain that \( \text{deg}(c(x)) \leq (n - k) + (k - 1) = n - 1 \), as expected.

**Remark:** the codes whose words can be represented as polynomials of the form

\[
\text{codeword polynomial} = \text{arbitrary polynomial} \times \text{fixed polynomial}
\]

are in general called cyclic codes.
Decoding Reed-Solomon codes

First, we give a brief overview of Reed-Solomon codes. Second, we introduce polynomials that will be useful in decoding of Reed-Solomon codes. The decoding process is based on operating with these polynomials. Third, we describe an algorithm for finding these polynomials.

Overview of Reed-Solomon codes

Consider an \([n, k, d]\) GRS code \(C\) over a finite field \(\mathbb{F}\) with the following parity-check matrix \(H\).

\[
H = \begin{pmatrix}
    v_1 & v_2 & \cdots & v_n \\
    v_1 \alpha_1 & v_2 \alpha_2 & \cdots & v_n \alpha_n \\
    v_1 \alpha_1^2 & v_2 \alpha_2^2 & \cdots & v_n \alpha_n^2 \\
    \vdots & \vdots & \ddots & \vdots \\
    v_1 \alpha_1^{d-2} & v_2 \alpha_2^{d-2} & \cdots & v_n \alpha_n^{d-2}
\end{pmatrix},
\]

where \(\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}\) are all distinct and nonzero (they are called code-locators). Additionally, \(v_1, v_2, \ldots, v_n \in \mathbb{F}\) are all nonzero and they are sometimes called column multipliers. From the Singleton bound, we have

\[
n - k = d - 1,
\]

and thus the matrix \(H\) has \(n - k\) rows.

Let \(\tilde{c} = (c_1, c_2, \ldots, c_n) \in C\) be a transmitted codeword and let \(\tilde{y} = (y_1, y_2, \ldots, y_n) \in \mathbb{F}^n\) be the received word that may be different from \(\tilde{c}\) due to errors during the transmissions. Let

\[
\tilde{e} = \tilde{y} - \tilde{c} = (e_1, e_2, \ldots, e_n)
\]

be the error vector. Since \(\tilde{c}, \tilde{y} \in \mathbb{F}^n\), we have that \(\tilde{e} \in \mathbb{F}^n\).

If there are no errors during the transmission of the codeword then the error vector contains only zeros; the nonzero entries in the error vector describe the errors. Therefore, the Hamming weight of the error vector shows the number of errors that occurred during the transmission of the codeword.

Let the set of error coordinates be denoted by \(J \subseteq \{1, 2, \ldots, n\}\). Therefore, for any \(i = 1, 2, \ldots, n\),

\[
i \in J \iff e_i \neq 0.
\]

We assume that the cardinality of \(J\) is \(|J| \leq \lfloor \frac{d-1}{2} \rfloor\), and therefore there is a unique codeword \(\tilde{c} \in C\) corresponding to the received \(\tilde{y} \in \mathbb{F}^n\).
Polynomials used for decoding

Next, we will define the syndrome polynomial. Recall, that a syndrome of a word $\bar{y} \in \mathbb{F}^n$ is defined in the following way

$$\bar{s}^T = \begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ \vdots \\ s_{d-2} \end{pmatrix} = H \cdot \bar{y}^T,$$

where $H$ is the parity-check matrix of $C$. If no errors occurred during the transmission then the syndrome will contain only zeros. To find a certain syndrome element, we have

$$s_\ell = \sum_{i=1}^{n} v_i \alpha_i^\ell \cdot y_i,$$

where $\ell = 0, 1, 2, \ldots, d-2$.

**Syndrome polynomial**

We define the syndrome polynomial as

$$S(x) = \sum_{\ell=0}^{d-2} s_\ell \cdot x^\ell.$$ 

Since $H \cdot \bar{y}^T = H \cdot e^T$, we can replace all appearances of $y_i$ with $e_i$ in the expression for $s_\ell$. After the replacement, we obtain

$$s_\ell = \sum_{i=1}^{n} v_i \alpha_i^\ell \cdot e_i = \sum_{j \in J} v_j \alpha_j^\ell \cdot e_j,$$

where the final sum contains only nonzero terms.

We can simplify the syndrome polynomial as follows:

$$S(x) = \sum_{\ell=0}^{d-2} s_\ell \cdot x^\ell$$

$$= \sum_{\ell=0}^{d-2} \left( \sum_{j \in J} v_j \alpha_j^\ell \cdot e_j \right) \cdot x^\ell$$

$$= \sum_{\ell=0}^{d-2} \left( \sum_{j \in J} v_j \alpha_j^\ell \cdot e_j \cdot x^\ell \right)$$

$$= \sum_{j \in J} v_j e_j \cdot \left( \sum_{\ell=0}^{d-2} \alpha_j^\ell \cdot x^\ell \right).$$
Next, we will simplify the sum
\[ \sum_{\ell=0}^{d-2} \alpha_j^\ell \cdot x^\ell . \]

By using an expression for geometric progression we obtain the equality
\[ (1 - \alpha_j x) \cdot \sum_{\ell=0}^{d-2} (\alpha_j x)^\ell = 1 - (\alpha_j x)^{d-1} . \]

Now we can take the modulo of \( x^{d-1} \) from both sides of the sum,
\[ (1 - \alpha_j x) \cdot \sum_{\ell=0}^{d-2} (\alpha_j x)^\ell \equiv 1 \mod x^{d-1} , \]
or,
\[ \sum_{\ell=0}^{d-2} (\alpha_j x)^\ell = \frac{1}{1 - \alpha_j x} \mod x^{d-1} . \]

Now, we can substitute this result into the expression for syndrome polynomial, in order to obtain
\[ S(x) = \sum_{j \in J} v_j e_j \cdot \frac{1}{1 - \alpha_j x} \mod x^{d-1} , \text{ if } x \neq \alpha_j^{-1} . \]

Error-locator and Error-evaluator polynomials

**Error-locator polynomial.** The degree of this polynomial is equal to the number of errors, \( |J| \).

The error-locator polynomial is defined by
\[ \Lambda(x) = \prod_{j \in J} (1 - \alpha_j x) . \]

**Error-evaluator polynomial.** The degree of this polynomial is \( |J| - 1 \). The error-evaluator polynomial is defined by
\[ \Gamma(x) = \sum_{j \in J} e_j v_j \cdot \prod_{m \in J \setminus \{j\}} (1 - \alpha_m x) . \]

**Properties of the polynomials.** The syndrome, error-locator and error-evaluator polynomials satisfy the following equation
\[ S(x) \cdot \Lambda(x) = \Gamma(x) \mod x^{d-1} . \] (1)
Decoding using polynomials

In order to decode the received vector $\tilde{y}$, the receiving party can compute the syndrome (and the syndrome polynomial $S(x)$). With the help of the syndrome polynomial, it is possible to find the other two polynomials, $\Lambda(x)$ and $\Gamma(x)$.

On one hand,
$$\Lambda(\alpha_i^{-1}) = 0 \iff i \in J.$$ 

On the other hand,
$$\Gamma(\alpha_i^{-1}) = e_i v_i \cdot \prod_{m \in J \setminus \{i\}} (1 - \alpha_m \alpha_i^{-1}).$$

The sets of roots of the polynomials $\Gamma(x)$ and $\Lambda(x)$ are disjoint:

$$\gcd(\Lambda(x), \Gamma(x)) = 1.$$ \hspace{1cm} (2)

We already know that $\deg \Lambda(x) = |J|$ and $\deg \Gamma(x) = |J| - 1$. Therefore, we can write that

$$\deg \Gamma(x) < \deg \Lambda(x) \leq \left\lfloor \frac{d-1}{2} \right\rfloor.$$ \hspace{1cm} (3)

The equations (1), (2), (3) together are called the key equation of decoding of GRS codes.

Peterson-Gorenstein-Zierler algorithm for finding the polynomials

Peterson-Gorenstein-Zierler algorithm is an easy (brute-force) way for finding of the polynomials $\Lambda(x)$ and $\Gamma(x)$.

Denote
$$\Lambda(x) = \sum_{i=0}^{\tau} \lambda_i \cdot x^i$$
and
$$\Gamma(x) = \sum_{i=0}^{\tau-1} \gamma_i \cdot x^i,$$
where $\tau$ denotes the maximum possible number of errors, $\tau = \left\lfloor \frac{d-1}{2} \right\rfloor$.

Construct a system of equations. If we know the syndrome polynomial, then we can solve this system and find the values of all $\lambda_i$ and $\gamma_i$. 
This system of linear equations with unknowns $\lambda_i$ and $\gamma_i$ over $\mathbb{F}$ is equivalent to equation (1). Solving of this system of equations consists of two steps. First, the last $d - \tau - 1$ equations would have to be solved in order to find the values of $\lambda_0, \lambda_1, \ldots, \lambda_\tau$. Then, the values of $\gamma_0, \gamma_1, \ldots, \gamma_{\tau-1}$ would have to be found by using the first $\tau$ equations.

**Theorem from the practice session**

Let $\mathcal{C}$ be an $[n, k, d]$ GRS code over $\mathbb{F}$.

**Theorem.** Let $\lambda(x)$ and $\gamma(x)$ be polynomials over $\mathbb{F}$ that satisfy $\deg(\gamma(x)) < \deg(\lambda(x)) \leq \frac{1}{2}(d-1)$. Then,

1. First,
   \[ \lambda(x) \cdot S(x) = \gamma(x) \mod x^{d-1} \]  
   if and only if there exists a polynomial $c(x)$ such that
   \[ \lambda(x) = c(x) \cdot \Lambda(x) \quad \text{and} \quad \gamma(x) = c(x) \cdot \Gamma(x) . \]

2. Second, the solution to (4) with a nonzero polynomial $\lambda(x)$ of the smallest possible degree is unique up to scaling by a nonzero constant $c \in \mathbb{F}$, such that
   \[ \lambda(x) = c \cdot \Lambda(x) \quad \text{and} \quad \gamma(x) = c \cdot \Gamma(x) . \]

3. Third, the solution in (5) satisfies
   \[ \gcd(\gamma(x), \lambda(x)) = 1 . \]
Reminder

- The syndrome polynomial: \( S(x) = s_{d-2}x^{d-2} + s_{d-1}x^{d-1} + \ldots + s_1x + s_0. \)
- The error-locator polynomial: \( \Lambda(x) = \prod_{j \in J} (1 - \alpha_j x). \)
- The error-evaluator polynomial: \( \Gamma(x) = \sum_{j \in J} e_j v_j \cdot \prod_{m \in J \setminus \{j\}} (1 - \alpha_m x), \) where \( J \) is the set of the coordinates in error.

The Key equations of decoding of GRS codes is given by the following three conditions:

1. \( \gcd(\Lambda(x), \Gamma(x)) = 1; \)
2. \( \deg(\Gamma(x)) < \deg(\Lambda(x)) \leq \frac{d-1}{2}; \)
3. \( \Lambda(x) \cdot S(x) = \Gamma(x) \pmod{x^{d-1}}. \)

In the previous lecture we saw Peterson-Gorenstein-Zierler decoding method. However, the decoding time complexity of that method, if executed in a straightforward manner, is cubic in \( n. \) In this lecture, we will study another decoding method, which uses Euclidean algorithm. The two methods differ in a way they solve the key equation.

Extended Euclid’s algorithm

Below, we present an extended version of Euclid’s algorithm. Let \( a(x) \) and \( b(x) \) be two given polynomials over the finite field \( \mathbb{F}. \) In addition to computation of \( \gcd(a(x), b(x)) \), the algorithm also finds polynomials \( s(x) \) and \( t(x) \) such that

\[
\gcd(a(x), b(x)) = a(x) \cdot s(x) + b(x) \cdot t(x).
\]

The formal description of the algorithm appears in Figure 1. For the last three equalities, the underlined polynomials are the ones that are computed in each of the corresponding rows.

Let \( \tau \) denote the largest index \( i \) such that \( r_i(x) \neq 0. \) Then, \( \gcd(a(x), b(x)) = r_{\tau}(x). \)

**Lemma.** The following conditions hold:

1. For \( i = -1, 0, 1, 2, \ldots, \tau + 1: r_i(x) = s_i(x) \cdot a(x) + t_i(x) \cdot b(x). \)
2. For \( i = 0, 1, 2, \ldots, \tau + 1: \deg(t_i(x)) + \deg(r_{i-1}(x)) = \deg(a(x)). \)
\[ r_{-1}(x) = a(x); \quad r_0(x) = b(x); \]
\[ s_{-1}(x) = 1; \quad s_0(x) = 0; \]
\[ t_{-1}(x) = 0; \quad t_0(x) = 1; \]
\[
\text{for ( } i = 1; \ r_{i-1}(x) \neq 0; \ i++ \text{ ) } \{
\begin{align*}
    r_{i-2}(x) &= q_i(x) \cdot r_{i-1}(x) + r_i(x); \\
    s_{i-2}(x) &= q_i(x) \cdot s_{i-1}(x) + s_i(x); \\
    t_{i-2}(x) &= q_i(x) \cdot t_{i-1}(x) + t_i(x);
\end{align*}
\}
\]

Figure 1: Extended Euclid’s algorithm.

**Proof.** The proof is by induction on \( i \). It is left for the homework.

**Remark.** Observe that for every iteration \( i \) in the algorithm, the degrees of the polynomials \( r_i(x) \) and \( t_i(x) \) satisfy

\[
\deg(r_{i-1}(x)) > \deg(r_i(x)) \quad \text{and} \quad \deg(t_{i-1}(x)) < \deg(t_i(x)),
\]

respectively.

**Lemma.** Suppose that \( t(x) \) and \( r(x) \) are nonzero polynomials over a finite field \( F \) satisfying the following conditions:

1. \( \gcd(t(x), r(x)) = 1; \)
2. \( \deg(t(x)) + \deg(r(x)) < a(x); \)
3. \( t(x) \cdot b(x) = r(x) \pmod{a(x)}. \)

Then, there exists an index \( h \in \{0, 1, \ldots, \tau + 1\} \) and constant \( c \in F \) such that

\[
t(x) = c \cdot t_h(x) \quad \text{and} \quad r(x) = c \cdot r_h(x).
\]  

(1)

**Proof.** The proof is omitted.

**Solving the key equations by Euclid’s algorithm**

We apply the algorithm in Figure 1 to the input \( a(x) = x^{d-1} \) and \( b(x) = S(x) \). We stop at iteration \( h \), and take \( \Lambda(x) = c \cdot t_h(x) \) and \( \Gamma(x) = c \cdot r_h(x) \), where \( c \in F \) is a constant, which is yet to be determined.

**Lemma.** Let \( t(x) \) and \( r(x) \) be as in the previous lemma. Additionally, assume that

\[
\deg(t(x)) \leq \frac{1}{2} \deg(a(x)) \quad \text{and} \quad \deg(r(x)) < \frac{1}{2} \deg(a(x)).
\]
Then, \( h \) is the unique index for which the remainders in the Euclid’s algorithm satisfy

\[
\deg(r_h(x)) < \frac{1}{2}(d - 1) \leq \deg(r_{h-1}(x)) \, .
\]  

(2)

**Corollary.** The solution to key equation is obtained by taking \( \Lambda(x) = c \cdot t_h(x) \) and \( \Gamma(x) = c \cdot r_h(x) \) for some nonzero constant \( c \in \mathbb{F} \), where \( \{t_i(x)\}_{i=-1}^{h} \) and \( \{r_i(x)\}_{i=-1}^{h} \) are obtained from the extended Euclid’s algorithm in Figure 1 applied to \( a(x) = x^{d-1} \) and \( b(x) = S(x) \), and where \( h \) is a unique value such that (2) holds.

**Formal derivative and Forney’s algorithm**

Let \( a(x) = \sum_{i=0}^{n} a_i x^i \) be a polynomial over a finite field \( \mathbb{F} \). Define its formal derivative as

\[
a'(x) = \sum_{i=1}^{n} i \cdot a_i x^{i-1} .
\]

The formal derivative obeys the following rules:

\[
(a(x) + b(x))' = a'(x) + b'(x) ,
\]

(3)

\[
(c \cdot a(x))' = c \cdot a'(x) ,
\]

(4)

\[
(a(x) \cdot b(x))' = a(x)' \cdot b(x) + a(x) \cdot b(x)' .
\]

(5)

Therefore, by repeating application of (5), we have

\[
\Lambda'(x) = \sum_{j \in J} (-\alpha_j) \cdot \prod_{m \in J \setminus \{j\}} (1 - \alpha_m x) .
\]

In particular, for all \( k \in J \), we obtain

\[
\Lambda'(\alpha_k^{-1}) = -\alpha_k \cdot \prod_{m \in J \setminus \{k\}} (1 - \alpha_m \alpha_k^{-1}) .
\]

Additionally, for all \( k \in J \),

\[
\Gamma(\alpha_k^{-1}) = e_k v_k \cdot \prod_{m \in J \setminus \{k\}} (1 - \alpha_m \alpha_k^{-1}) .
\]

By summing up these two results, we have that

\[
e_k = -\frac{\alpha_k}{v_k} \cdot \frac{\Gamma(\alpha_k^{-1})}{\Lambda'(\alpha_k^{-1})} .
\]

This method of computing of error values is called Forney’s algorithm.