Chapter 10 Error-Control Coding

Error-Control Coding is the platform of Shannon channel coding theorem.

10.1 Introduction
10.1 Introduction

- Error correction versus error detection
  - There is an alternative system approach to achieve reliable transmission other than forward error correction (FEC).
  - By the system structure, it is named automatic-repeat request (ARQ), which is a combination of error detection and noiseless feedback.

10.1 Introduction

- Classifications of ARQ
  - ARQ with stop-and-wait strategy
    - After the transmission of a codeword, the transmitter stops and waits for the feedback before moving onto the next block of message bits.
  - Continuous ARQ with pullback
    - The transmitter continue its transmissions until a retransmission request is received, at which point it stops and pulls back to the incorrectly transmitted codeword.
  - Continuous ARQ with selective repeat
    - Only retransmit the codewords that are incorrectly transmitted.
10.3 Linear block codes

“10.2 Discrete memoryless channels” has been introduced in Chapter 9, so we omit it.

Linear code

- A code is linear if any two code words in the code can be added in modulo-2 arithmetic to produce a third code word in the code.
- The codewords of a linear code can always be obtained through a “linear” operation in the sense of modulo-2 arithmetic.

For a linear code, there exists a $k$-by-$n$ generator matrix $G$ such that

$$c_{1 \times n} = m_{1 \times k}G_{k \times n}$$

where code bits $c = [c_0, c_1, \ldots, c_{n-1}]$

message bits $m = [m_0, m_1, \ldots, m_{k-1}]$

(Here, we use modulo-2 addition and modulo-2 multiplication.)

Generator matrix $G$ is said to be in the canonical form if its $k$ rows are linearly independent.
What will happen if one row is linearly dependent on other rows?

\[
\begin{bmatrix}
c_0 & \cdots & c_{n-1}
\end{bmatrix} =
\begin{bmatrix}
m_0 & \cdots & m_{k-1}
\end{bmatrix}
\begin{bmatrix}
g_{0,0} & g_{0,1} & \cdots & g_{0,n-1}
g_{1,0} & g_{1,1} & \cdots & g_{1,n-1}
g_{2,0} & g_{2,1} & \cdots & g_{2,n-1}
\vdots & \vdots & \ddots & \vdots
g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1}
\end{bmatrix}
\]

Suppose

\[a \begin{bmatrix} g_{0,0} & g_{0,1} & \cdots & g_{0,n-1} \end{bmatrix} + b \begin{bmatrix} g_{1,0} & g_{1,1} & \cdots & g_{1,n-1} \end{bmatrix} = \begin{bmatrix} g_{2,0} & g_{2,1} & \cdots & g_{2,n-1} \end{bmatrix}.
\]

Then

\[
\begin{bmatrix}
c_0 & \cdots & c_{n-1}
\end{bmatrix} =
\begin{bmatrix}
m_0 & \cdots & m_{k-1}
\end{bmatrix}
\begin{bmatrix}
g_{0,0} & g_{0,1} & \cdots & g_{0,n-1}
g_{1,0} & g_{1,1} & \cdots & g_{1,n-1}
ag_{0,0} + bg_{1,0} & ag_{0,1} + bg_{1,1} & \cdots & ag_{0,n-1} + bg_{1,n-1}
\vdots & \vdots & \ddots & \vdots
g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1}
\end{bmatrix}
\]

Hence, the number of distinct code words is at most \(2^{k-1}\) (not the anticipated \(2^k\)).

\[
\begin{bmatrix}
c_0 & \cdots & c_{n-1}
\end{bmatrix}
= \begin{bmatrix}
m_0 & m_1 & \cdots & m_{k-1}
\end{bmatrix}
\begin{bmatrix}
g_{0,0} & g_{0,1} & \cdots & g_{0,n-1}
g_{1,0} & g_{1,1} & \cdots & g_{1,n-1}
ag_{0,0} + bg_{1,0} & ag_{0,1} + bg_{1,1} & \cdots & ag_{0,n-1} + bg_{1,n-1}
\vdots & \vdots & \ddots & \vdots
g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1}
\end{bmatrix}_{k \times n}
\]

\[
= \begin{bmatrix}
m_0 & m_1 & \cdots & m_{k-1}
\end{bmatrix}
\begin{bmatrix}
g_{0,0} & g_{0,1} & \cdots & g_{0,n-1}
g_{1,0} & g_{1,1} & \cdots & g_{1,n-1}
g_{2,0} & g_{2,1} & \cdots & g_{2,n-1}
\vdots & \vdots & \ddots & \vdots
g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1}
\end{bmatrix}_{(k-1) \times n}
\]

where \(\hat{m}_0 = m_0 + m_2a\) and \(\hat{m}_1 = m_1 + m_2b\).

Hence, the number of distinct code words is at most \(2^{k-1}\) (not the anticipated \(2^k\)).
10.3 Linear block codes

- **Parity-check matrix** \( H \)
  - The parity-check matrix of a canonical generator matrix is an \((n-k)\)-by-\(n\) matrix satisfying
    \[
    H_{(n-k)\times n} G_{n\times k}^T = 0_{(n-k)\times k}
    \]
    where the columns of \( H \) are linearly independent.
  - Then, the code words (or error-free receptions) should satisfy \((n-k)\) parity-check equations.
    \[
    \Rightarrow c_{1\times n} H_{n\times(n-k)}^T = m_{1\times k} G_{k\times n} H_{n\times(n-k)}^T = 0_{1\times(n-k)}
    \]

- **Syndrome** \( s \)
  - The receptions may be erroneous (with error pattern \( e \)).
    \[
    r = c + e, \quad \text{“+” \equiv exclusive or.}
    \]
    \[
    e_i = \begin{cases} 
  1, & \text{if an error has occurred at the } i\text{th location;} \\
  0, & \text{otherwise} 
\end{cases}
    \]
  - With the help of parity-check matrix, we obtain
    \[
    s_{1\times(n-k)} = r_{1\times n} H_{n\times(n-k)}^T = c_{1\times n} H_{n\times(n-k)}^T + e_{1\times n} H_{n\times(n-k)}^T
    \]
    \[
    = e_{1\times n} H_{n\times(n-k)}^T
    \]
10.3 Linear block codes

- In short, syndromes are all possible symptoms that possibly happen at the output of parity-check examination.
  - Similar to the disease symptoms that can be observed and examined from outside the body.
  - Based on the symptoms, the doctors diagnose (possibly) what disease occurs inside the body.
- Based on the symptoms, the receiver “diagnoses” which bit is erroneous (i.e., ill) based on the symptoms (so that the receiver can correct the “ill” bit.)
  - Notably, the syndrome only depends on the error pattern, and is completely independent of the transmitted code word.

---

10.3 Linear block codes

- Properties of syndromes
  - Syndrome depends only on the error pattern, and not on the transmitted code word.
    \[
    s_{1 \times (n-k)} = r_{1 \times n} H_{n \times (n-k)}^T = c_{1 \times n} H_{n \times (n-k)}^T + e_{1 \times n} H_{n \times (n-k)}^T = e_{1 \times n} H_{n \times (n-k)}^T
    \]
  - All error patterns that differ by (at least) a code word have the same syndrome $s$.
    \[
    \text{coset}(s_{1 \times (n-k)}) = \{ e_{1 \times n} + c_{1 \times n} : \text{for some codeword } c_{1 \times n} \text{ and for some error pattern } e_{1 \times n} \text{ satisfying } s_{1 \times (n-k)} = e_{1 \times n} H_{n \times (n-k)}^T \}
    \]
0. It suffices to fix the error pattern \( e \) and vary the codewords when determining the coset.

1. All elements in a coset have the same syndrome since 
   \[ cH^T = 0 \]

2. There are \( 2^k \) elements in a coset since
   \[ e + c_i = e + c_j \Leftrightarrow c_i = c_j \]
   i.e., coset elements are the same iff two code words are the same.

3. Cosets are disjoint.

4. The number of cosets is \( 2^{n-k} \), i.e., the number of syndromes is \( 2^{n-k} \).
   Thus, syndromes (with only \( n-k \) unknowns) cannot uniquely determine the error pattern (with \( n \) unknowns).

### 10.3 Linear block codes

- **Systematic code**
  - A code is systematic if the message bits are a part of the codewords.
  - The remaining part of a systematic code is called *parity bits*.

  \[
  c_i = \begin{cases} 
  b_i, & i = 0, 1, \ldots, n - k - 1 \\
  m_{i-(n-k)}, & i = n - k, n - k + 1, \ldots, n - 1
  \end{cases}
  \]

  Usually, message bits are transmitted first because the receiver can do “direct hard decision” when “necessary”.
10.3 Linear block codes

- For a systematic code,
  
  \[ G = \begin{bmatrix} P_{k \times (n-k)} & I_k \end{bmatrix} \quad \text{and} \quad H = \begin{bmatrix} I_{n-k} & P_{(n-k) \times k}^T \end{bmatrix}, \]

  where \( I_k \) is the \( k \)-by-\( k \) identity matrix.

- Example 10.1. \((n, 1)\) repetition codes
  
  \[ P_{1 \times (n-1)} = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \]

---

10.3 Linear block codes

- Error correcting capability of a linear block code
  
  - Hamming distance (for “0”-“1” binary sequences)
    
    \[ d_H(u, v) = \sum_{i=1}^{n} (u_i + v_i), \quad “+” \equiv \text{exclusive or}. \]

  - Minimum (pair-wise) Hamming distance \( d_{\text{min}} \)
    
    \[ d_{\text{min}} = \min_{e_i, e_j \in C, \ e_i \neq e_j} d_H(e_i, e_j) \]
 Operational meaning of the minimum (pair-wise) Hamming distance

There exists at least a pair of codewords, of which the distance is $d_{\text{min}}$.

Minimum distance decoder

$$\text{decision} = \arg \min_{c \in \mathcal{C}} d_H(r, c)$$

Based on the minimum distance decoder, if the received vector and the transmitted codeword differ at most

$$\left\lfloor \frac{1}{2} (d_{\text{min}} - 1) \right\rfloor$$

namely, if the number of 1’s in the (true) error pattern is at most this number, then no error in decoding decision is obtained.

If the received vector and the transmitted codeword differ at $t$ positions, where

$$t > \left\lfloor \frac{1}{2} (d_{\text{min}} - 1) \right\rfloor$$

then erroneous decision is possibly made (such as when the transmitted codeword is one of the pairs with distance $d_{\text{min}}$).
We therefore name this quantity the **error correcting capability** of a code (not limited to linear block codes).

\[
\frac{1}{2}(d_{\text{min}} - 1)
\]

The **error correcting capability** of a **linear** block code can be easily determined by the minimum **Hamming weight** of codewords. (This does not apply for **non-linear** codes!)

\[
w_H(u) = d_H(u, 0) = \sum_{i=1}^{n} (u_i + 0) = \sum_{i=1}^{n} u_i
\]

By linearity,

\[(\forall \ c_i \text{ and } c_j \in C) \text{ there exists } c_k \in C \text{ such that } w_H(c_k) = d_H(c_i, c_j)\]

\[\Rightarrow d_{\text{min}} = \min_{c_i, c_j \in C, c_i \neq c_j} d_H(c_i, c_j) = \min_{c \in C} w_H(c)\]

---

**10.3 Linear block codes**

- Syndrome decoding for \((n, k)\) linear block codes

decision \[\Rightarrow \]

\[
\begin{align*}
\text{decision} & = \arg \min_{e \in C} d_H(r, c) \\
& = \arg \min_{e \in C} d_H(r + c, 0) \Rightarrow \arg \min_{e + r \in C} d_H(r + e + r, 0) \\
& = \arg \min_{e \in C} d_H(e, 0) + r \\
& = \min_{e \in \coset(rH^T)} w_H(e) + r \\
& = e + e + c \Rightarrow \\
& \begin{cases} 
  e = r + c \\
  c = e + r
\end{cases}
\end{align*}
\]

---

© Po-Ning Chen@cm.nctu

Chapter 10-19
10.3 Linear block codes

- Syndrome decoding using standard array for an \((n, k)\) block code

\[
\begin{array}{cccccc}
& c_1 = 0 & c_2 & c_3 & \ldots & c_r & \ldots & c_{2^k} \\
\hline
e_2 & c_2 + e_2 & c_3 + e_2 & \ldots & c_r + e_2 & \ldots & c_{2^k} + e_2 \\
e_3 & c_2 + e_3 & c_3 + e_3 & \ldots & c_r + e_3 & \ldots & c_{2^k} + e_3 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots
\end{array}
\]

Syndrome \(rH\)

- Find the element in coset \((rH)\), whose has the minimum weight.
- This element is usually named the coset leader.
- The coset leader in each coset is fixed and known before the reception of \(r\).
10.3 Linear block codes

- Decoding table

<table>
<thead>
<tr>
<th>syndrome</th>
<th>coset leader</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0000000</td>
</tr>
<tr>
<td>100</td>
<td>1000000</td>
</tr>
<tr>
<td>010</td>
<td>0100000</td>
</tr>
<tr>
<td>001</td>
<td>0010000</td>
</tr>
<tr>
<td>110</td>
<td>0001000</td>
</tr>
<tr>
<td>011</td>
<td>0000100</td>
</tr>
<tr>
<td>111</td>
<td>0000010</td>
</tr>
<tr>
<td>101</td>
<td>0000001</td>
</tr>
</tbody>
</table>

\[ r = [1100010] \]
\[ \Rightarrow rH^T = [001] \]
\[ \Rightarrow e = [0010000] \]
\[ \Rightarrow c = r + e = [1100010] + [0010000] = [1110010] \]

Appendix: The notion of perfect code

- (7, 4) Hamming code is a binary perfect code.

For (7, 4) Hamming code, the coset leaders form a perfect ball (of radius 1).

All of the \(2^7 = 128\) binary sequences are confined with the \(2^4 = 16\) non-overlapping balls of radius 1,

\[ \text{i.e.} \sum_{t=0}^{1} \binom{7}{t} = \frac{2^7}{2^4} = 2^3. \]
10.3 Linear block codes

- **Dual code**

  \[ H_{(n-k) \times n} G_{n \times k}^T = 0_{(n-k) \times k} \]

  \[ \Rightarrow G_{k \times n} H_{n \times (n-k)}^T = H_{k \times n} G_{n \times (n-k)}^T = 0_{k \times (n-k)}. \]

- Every \((n, k)\) linear block code with generator matrix \(G\) and parity-check matrix \(H\) has an \((n, n-k)\) dual code with generator matrix \(H\) and parity-check matrix \(G\).

10.4 Polynomial codes/Cyclic codes

- **Polynomial expression of a linear code**

  - **Polynomial code:** A special type of linear codes.

    - Represent a code \([c_0, c_1, \ldots, c_{n-1}]\) as a code polynomial of degree \(n-1\)
      
      \[ c(X) = c_0 + c_1 X + c_2 X^2 + \cdots + c_{n-1} X^{n-1} \]

      where \(X\) is called the indeterminate.
10.4 Polynomial codes/Cyclic codes

- Generator polynomial (of a polynomial code)

\[ g(X) = 1 + g_1X + g_2X^2 + \cdots + g_{n-k-1}X^{n-k-1} + X^{n-k} \]

- The code polynomial (of a polynomial code) includes all polynomials of degree \((n-1)\), which can be divided by \(g(X)\), and hence can be expressed as

\[ c(X) = a(X)g(X) \]

---

**Example of a \((6, 3)\) polynomial code**

\[ c(X) = (a_0 + a_1X + a_2X^2)(1 + g_1X + g_2X^2 + X^3) \]

\[
\begin{bmatrix}
  c_0 & c_1 & c_2 & c_3 & c_4 & c_5 \\
  a_0 & a_1 & a_2 \\
  g_1 & g_2 & 1 & 0 & 0 \\
  0 & 1 & g_1 & g_2 & 1 & 0 \\
  0 & 0 & 1 & g_1 & g_2 & 1
\end{bmatrix}
\]

So, the polynomial code is a special type of linear codes.
10.4 Polynomial codes/Cyclic codes

- Property of a polynomial code

Let \( c(X) \) be the code polynomial corresponding to \( a(X) \), i.e., \( c(X) = a(X)g(X) \).

\[ X^{n-k}a(X) = q(X) \cdot g(X) + r(X), \]
where the degree of remainder \( r(X) \) is less than \( n - k \).

\[ \bar{c}(X) = X^{n-k}a(X) - r(X) = q(X)g(X) \] is an alternative way to represent code polynomials.

- The last \( k \) bits of \( \bar{c}(X) \) should be exactly the same as \( a(X) \).
As a result, \( \bar{c}(X) \) is a systematic code.

---

10.4 Polynomial codes/Cyclic codes

- Example of a \((6, 3)\) polynomial code (Continue)

<table>
<thead>
<tr>
<th>( q_1q_2 )</th>
<th>( \bar{c}_0\bar{c}_1\bar{c}_2\bar{c}_3\bar{c}_4\bar{c}_5 )</th>
<th>( a_0a_1a_2 )</th>
<th>( c_0c_1c_2c_3c_4c_5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>000000</td>
<td>000</td>
<td>000000</td>
</tr>
<tr>
<td>011</td>
<td>010001</td>
<td>001</td>
<td>001111</td>
</tr>
<tr>
<td>110</td>
<td>100010</td>
<td>010</td>
<td>011110</td>
</tr>
<tr>
<td>101</td>
<td>110110</td>
<td>011</td>
<td>010001</td>
</tr>
<tr>
<td>100</td>
<td>111100</td>
<td>100</td>
<td>111100</td>
</tr>
<tr>
<td>111</td>
<td>101101</td>
<td>101</td>
<td>110101</td>
</tr>
<tr>
<td>010</td>
<td>011110</td>
<td>110</td>
<td>100010</td>
</tr>
<tr>
<td>001</td>
<td>001111</td>
<td>111</td>
<td>101101</td>
</tr>
</tbody>
</table>

\[
\bar{c}(X) = X^3(a_0 + a_1X + a_2X^2) - X^3(a_0 + a_1X + a_2X^2) \mod (1 + g_1X + g_2X^2 + X^3)
\]

\[
[\bar{c}_0 \ \bar{c}_1 \ \bar{c}_2 \ \bar{c}_3 \ \bar{c}_4 \ \bar{c}_5] = [a_0 \ a_1 \ a_2] \begin{bmatrix} 1 & g_1 & g_2 & 1 & 0 & 0 \\ g_2 & 1 + g_1g_2 & g_1 + g_2 & 0 & 1 & 0 \\ g_1 + g_2 & g_1 + g_2 + g_1g_2 & 1 + g_2 & 0 & 0 & 1 \end{bmatrix}
\]

So, the polynomial code can be made equivalent to a systematic linear code.
10.4 Polynomial codes/Cyclic codes

Encoder for systematic polynomial codes

\[
\bar{c}(X) = X^{n-k}a(X) + r(X)
\]

\[
\begin{align*}
\bar{c}(X) &= X^{n-k}a(X) + r(X) \\
&= \frac{X^{n-k}a(X)}{\text{degrees } n-k \text{ to } n-1} + \left[ \frac{X^{n-k}a(X) \mod g(X)}{\text{degrees } 0 \text{ to } n-k-1} \right] \\
&= \left[ \bar{c}_0 \cdots \bar{c}_{n-k-1} \bar{c}_{n-k} \cdots \bar{c}_{n-1} \right] = [u_0 \cdots u_{n-k-1} a_0 \cdots a_{k-1}] \\
\end{align*}
\]

How to find the remainder \((u_0 \ u_1 \cdots \ u_{n-k-1})\) of \(X^{n-k}a(X)\) divided by \(g(X)\)?

\[
X^{n-k}a(X) = g(X)q(X) + u(X)
\]

Example of the usual long division

\[X^{13} + X^{11} + X^{10} + X^7 + X^3 + X + 1 \text{ divided by } X^6 + X^5 + X^4 + X^3 + 1\]
\[ X^6 + X^3 + X^3 + X^3 + 1 \]

\[ X^{13} + X^{11} + X^{10} + X^7 + X^4 + X^3 + X + 1 \]

\[ X^{12} + X^{11} + X^{10} + X^7 + 0 + 0 + X^4 + X^3 + 0 + X + 1 \]

\[ X^{12} + X^{11} + X^{10} + X^9 + 0 + 0 + X^6 \]

\[ X^{11} + X^{10} + X^9 + 0 + X^6 \]

\[ \ldots \] repeat this procedure until the last term is shifted into the shift register.

- **Temporary remainder**
- **Temporary quotient**
An alternative realization of the long division (with the shift register containing not the remainder but the “lower power coefficients”)

\[
X^7(X^6 + X^5 + X^4 + 1) = X^{13} + X^{12} + X^{11} + X^{10} + 0 + 0 + + 0 + X^7
\]

[Diagram showing a shift register with the coefficients and the division process, including the polynomial expressions and the resulting quotient and remainder.]
On for first $k$ clocks
Off for the remaining $(n-k)$ clocks

After $k$ clocks,

$q(X)g(X) = \bar{c}(X) = a_{k-1}X^{n-1} + \cdots + a_0X^{n-k} + u_{n-k-1}X^{n-k-1} + \cdots + u_0$
10.4 Polynomial codes/Cyclic codes

Decoder for polynomial codes

- How to find the syndrome polynomial \( s(X) \) of polynomial codes with respect to the received word polynomial \( r(X) \)?
- Recall that syndromes are all symptoms that possibly happen at the output of parity-check examination.
- If there is no error in transmission, \( r(X) \mod g(X) = 0 \). Indeed, \( s(X) = r(X) \mod g(X) \).

\[ X^{n-k}a(X) = q(X)g(X) + u(X) \]

10.4 Polynomial codes/Cyclic codes

- Relation of syndrome polynomial \( s(X) \), error polynomial \( e(X) \) and received vector polynomial \( r(X) \) for systematic polynomial codes.

\[
\begin{align*}
    r(X) &= q_r(X)g(X) + s(X) \\
    \text{Also, } r(X) &= \bar{e}(X) + e(X) \\
    \text{where } e(X) \text{ is the error polynomial} \\
    &= q(X)g(X) + e(X) \\
    \Rightarrow s(X) &= r(X) \mod g(X) = e(X) \mod g(X) \\
    \Rightarrow e(X) &= q_e(X)g(X) + s(X)
\end{align*}
\]
10.4 Polynomial codes/Cyclic codes

- Relation of syndrome polynomial $s(X)$, error polynomial $e(X)$ and received vector polynomial $r(X)$ for general polynomial codes.

$$r(X) = q_c(X)g(X) + s(X)$$

Also, $r(X) = e(X) + e(X)$ where $e(X)$ is the error polynomial

$$= a(X)g(X) + e(X)$$

$$\Rightarrow s(X) = r(X) \mod g(X) = e(X) \mod g(X)$$

$$\Rightarrow e(X) = q_c(X)g(X) + s(X)$$
10.4 Polynomial codes/Cyclic codes

Example of a (6, 3) polynomial code (Continue)

\[ \bar{c}(X) = X^3(a_0 + a_1 X + a_2 X^2) \]
\[ -X^3(a_0 + a_1 X + a_2 X^2) \mod (1 + g_1 X + g_2 X^2 + X^3) \]
\[ [\bar{c}_0 \bar{c}_1 \bar{c}_2 \bar{c}_3 \bar{c}_4 \bar{c}_5] = [a_0 \ a_1 \ a_2] \begin{bmatrix} 1 & g_1 & g_2 & 1 & 0 & 0 \\ g_2 & 1 & g_1 g_2 & g_1 + g_2 & 0 & 0 \\ g_1 + g_2 & g_1 + g_2 + g_1 g_2 & 1 + g_2 & 0 & 0 & 1 \end{bmatrix} \]

So, the polynomial code can be made equivalent to a systematic linear code.
10.4 Polynomial codes/Cyclic codes

- The decoding of a systematic code is to simply add the coset leader, corresponding to the syndrome polynomial, to the received vector polynomial.

Definition of cyclic codes

- **Cyclic property**: Any cyclic shift of a codeword is also a codeword.
- **Linearity property**: Any sum of two codewords is also a codeword.

A cyclic code is also a polynomial code.

10.4 Polynomial codes/Cyclic codes

- A **cyclic code** is a special type of polynomial codes with $g(X)$ dividing $(X^n+1)$.

- Proof: Suppose $c(X)$ is a (non-zero) code polynomial.

$$X^ic(X) = X^i(c_0 + c_1X + \cdots + c_{n-1}X^{n-1})$$
$$= c_0X^i + c_1X^{i+1} + \cdots + c_{n-1}X^{n+i-1}$$
$$= c_{n-i} + c_{n-i+1}X + \cdots + c_{n-1}X^{i-1} + c_0X^i + c_1X^{i+1} + \cdots + c_{n-1}X^{n-1} + c_nX^n + 1 + c_{n-i+1}X(X^n + 1) + \cdots + c_{n-1}X^{i-1}(X^n + 1)$$
$$= c^{(i)}(X) + q^{(i)}(X)(X^n + 1)$$

where $c^{(i)}(X)$ is a codeword due to cyclic property.

\[\begin{align*}
q^{(i)}(X)(X^n + 1) &= X^ic(X) - c^{(i)}(X) \\
&= X^i(a(X)g(x) - a^{(i)}(X)g(X) \\
&= [X^ia(X) - a^{(i)}(X)]g(X)
\end{align*}\]

where $q^{(i)}(X) = c_{n-i} + c_{n-i+1}X + \cdots + c_{n-1}X^{i-1}$.

\[
\begin{aligned}
q^{(1)}(X)(X^n + 1) &= [Xa(X) - a^{(1)}(X)]g(X) \\
q^{(2)}(X)(X^n + 1) &= [X^2a(X) - a^{(2)}(X)]g(X) \\
&\vdots \\
q^{(n-1)}(X)(X^n + 1) &= [X^{n-1}a(X) - a^{(2)}(X)]g(X)
\end{aligned}
\]

\[\Rightarrow g(X) \text{ must divide } (X^n + 1).\]

A polynomial code is cyclic iff its generator polynomial divides $X^n+1$. All $(n-1)$ equations must be satisfied, and at least one $q^{(i)}(X)$ is non-zero.
10.4 Polynomial codes/Cyclic codes

Parity-check polynomial of cyclic codes

- After proving that \( g(X) \) must divide \( X^n+1 \), we can define the parity-check polynomial of a cyclic code as

\[
g(X)h(X) \mod (X^n + 1) = 0
\]

where \( h(X) \) is a polynomial of degree \( k \) with \( h_0 = h_k = 1 \).

- Since the degrees of \( g(X) \) and \( h(X) \) are respectively \( n-k \) and \( k-1 \), and \( g_{n-k} = h_k = 1 \), we may induce that

\[
g(X)h(X) = X^n + 1
\]

Multiplying both sides by \( a(X) \) yields:

\[
a(X)g(X)h(X) = c(X)h(X)
\]

\[
= \left( \sum_{\text{degrees } n \text{ to } n+k-1} X^n a(X) \right) + \left( \sum_{\text{degrees } 0 \text{ to } k-1} a(X) \right)
\]

⇒ Coefficients of \( c(X)h(X) \) equal zeros for degrees \( k \) to \( n-1 \)

\[
\Rightarrow \sum_{i=j}^{j+k} c_i h_{j+i} = 0 \text{ for } 0 \leq j \leq n - k - 1
\]
\[
\begin{bmatrix}
1 & 0 & 0 & \cdots & 0 \\
h_{k-1} & 1 & 0 & \cdots & 0 \\
h_{k-2} & h_{k-1} & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
h_1 & h_2 & h_3 & \cdots & 0 \\
1 & h_1 & h_2 & \cdots & 0 \\
0 & 1 & h_1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
\end{bmatrix}_{n \times (n-k)}
\]

\[c_0 \ c_1 \ \cdots \ c_{n-1}\]

\[c_1 \times n \mathbf{H}^T_{n \times (n-k)} = \mathbf{0}_{1 \times (n-k)}
\]
Recall that \( G_{k \times n} = \begin{bmatrix} 1 & g_1 & g_2 & \cdots & g_{n-k-1} & 1 & 0 & \cdots & 0 \\ 0 & 1 & g_1 & \cdots & g_{n-k-2} & g_{n-k-1} & 1 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & g_{n-k-3} & g_{n-k-2} & g_{n-k-1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & g_{n-2k} & g_{n-2k+1} & \cdots & \cdots & 1 \end{bmatrix}_{k \times n} \)

Also recall that \( G_{k \times n} H^T_{n \times (n-k)} = 0_{k \times (n-k)} \).

**Observation.**

The parity-check matrix arranges its entries according to the coefficients of the parity-check polynomial in reverse order as contrary to the generator matrix arranges its entries according to the coefficients of the generator polynomial.

### 10.4 Polynomial codes/Cyclic codes

**Remarks**

- The generator matrix and parity-check matrix derived previously are not for systematic codes.
- We can manipulate these two matrices by adding their elements of selective rows so as to make them “systematic”.
- Example can be found in Examples 10.3 and 10.4 in the textbook.
10.4 Polynomial codes/Cyclic codes

- Examples 10.3 and 10.4: Hamming code and Maximum-length code

\[ X^7 + 1 = (X^3 + X^2 + 1)(X^3 + X^2 + 1)(X + 1) \]

- Irreducible polynomial: A polynomial that cannot be further factored.
- All three of the above are irreducible.
- Primitive polynomial: An irreducible polynomial of degree \( m \), which divides \( X^n + 1 \) for \( n = 2^m - 1 \) but does not divides \( X^n + 1 \) for \( n < 2^m - 1 \).
- Only the first two irreducible polynomials are primitive.

Example 10.3 (of cyclic codes): \((7, 4, 3)\) Hamming code

- Any cyclic code generated by a primitive polynomial is a Hamming code of minimum pairwise distance 3.

Example 10.4 (of cyclic codes): \((2^m - 1, m, 2^m - 1)\) maximum-length code

- The maximum-length code is a dual code of Hamming codes.
- In other words, it is a cyclic code with primitive parity-check polynomial.
- It is a code of minimum distance \( 2^m - 1 \).
It is named the **maximum-length code** because the codeword length for \( m \) information bits has been pushed to the maximum (or equivalently, the number of codewords for code of length \( n \) has been pushed to the minimum). For example,

\[
\begin{align*}
\text{if } (c_0, c_1, \ldots, c_6) \text{ is a codeword, then} \\
(c_0, c_1, \ldots, c_6) \\
(c_1, c_2, \ldots, c_0) \\
(c_2, c_3, \ldots, c_1) \quad \text{are all codewords,} \\
\vdots \\
(c_6, c_0, \ldots, c_5)
\end{align*}
\]

So, there are in total 7 code words if \( c \) is originally nonzero. Adding the all-zero codeword gives \( 8 = 2^3 \) codewords. This makes the \((7, 3)\) maximum-length code.

So, the nonzero codeword in a maximum-length code must be a circular shift of any other nonzero codeword.
Example (of cyclic codes): Cyclic redundancy check (CRC) codes

- Cyclic codes are extremely well-suited for error detection owing to the simplicity of its implementation, and its superior error-detection capability.

- Binary \((n, k, d_{\text{min}})\) CRC codes can detect:
  - all contiguous error bursts of length \(n - k\) or less.
  - \(2^{n-k+1} - 4\) of contiguous error bursts of length \(n - k + 1\)
  - all combinations of \(d_{\text{min}} - 1\) or fewer errors
  - all error patterns with an odd number of errors if the generator polynomial \(g(X)\) has an even number of nonzero coefficients.

<table>
<thead>
<tr>
<th>Code</th>
<th>Generator Polynomial (g(X))</th>
<th>(n - k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC-12 code</td>
<td>(1 + X + X^2 + X^3 + X^{11} + X^{12})</td>
<td>12</td>
</tr>
<tr>
<td>CRC-16 code</td>
<td>(1 + X^2 + X^{15} + X^{16})</td>
<td>16</td>
</tr>
<tr>
<td>CRC-ITU code</td>
<td>(1 + X^5 + X^{12} + X^{16})</td>
<td>16</td>
</tr>
</tbody>
</table>

Example (of cyclic codes): Bose-Chaudhuri-Hocquenghem (BCH) codes

- A special type of BCH codes: Primitive \((n, k, d_{\text{min}})\) BCH codes with parameters satisfying

\[
\begin{align*}
  n & = 2^m - 1 \\
  k & \geq n - mt \\
  d_{\text{min}} & \geq 2t + 1 \\
  m & \geq 3 \\
  t & \leq (2^m - 1)/2
\end{align*}
\]

- \((n, k, 3)\) Hamming code can be described as BCH codes.

- The table in the next slide illustrates the generator polynomial \(g(X)\) of some binary BCH codes.
<table>
<thead>
<tr>
<th>$n$</th>
<th>$k$</th>
<th>$t$</th>
<th>Generator Polynomial</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>4</td>
<td>1</td>
<td>1 011</td>
</tr>
<tr>
<td>15</td>
<td>11</td>
<td>1</td>
<td>10 011</td>
</tr>
<tr>
<td>15</td>
<td>7</td>
<td>2</td>
<td>111 010 001</td>
</tr>
<tr>
<td>15</td>
<td>5</td>
<td>3</td>
<td>10 100 110 111</td>
</tr>
<tr>
<td>31</td>
<td>26</td>
<td>1</td>
<td>100 101</td>
</tr>
<tr>
<td>31</td>
<td>21</td>
<td>2</td>
<td>11 101 101 001</td>
</tr>
<tr>
<td>31</td>
<td>16</td>
<td>3</td>
<td>1 000 111 110 101 111</td>
</tr>
<tr>
<td>31</td>
<td>11</td>
<td>5</td>
<td>101 100 010 011 011 010 101</td>
</tr>
<tr>
<td>31</td>
<td>6</td>
<td>7</td>
<td>11 001 011 011 110 101 000 100 111</td>
</tr>
</tbody>
</table>

$n = \text{block length}$
$k = \text{number of message bits}$
$t = \text{number of errors that are guaranteed correctable}$

\[ X^5 + X^7 + X^6 + X^4 + 1 \]

**10.5 Convolutional codes**

- For block codes, the encoding and decoding must perform in a block-by-block basis. Hence, a big buffer for the entire message block and codeword block is required.
- Instead, for the convolutional codes, since the inputs and outputs are governed through a convolution operation of “finite-order” filter, only a buffer of size equal to the filter order is demanded.
  - “Convolution” is defined based on modulo-2 arithmetic operations.
10.5 Convolutional codes

Buffer two bits at a time

\((n, k, m) = (2, 1, 2)\) convolutional code

Note that here, \(n\) is not the codeword length, \(k\) is not the message length, and \(m\) is not the minimum distance between codewords. See the next slide for their definitions.

- \((n, k, m)\) convolutional codes
- \(n\) input bits will produce \(n\) output bits.
- The most recent \(m\) “\(k\)-message-bit input blocks” will be recorded (buffered).
- The \(n\) output bits will be given by a linear combination of the buffered input bits.
- Constraint length \(K\) of a convolutional code
  - It is the number of \(k\)-message-bit shifts, over which a single \(k\)-message-bit block influences the encoder output.
  - In other words, the encoder output depends on the current input message block and the previous \(K - 1\) input message blocks.
  - Based on the definition, constraint length = \(m+1\).
10.5 Convolutional codes

- **Effective code rate** of a convolutional code
  - In practice, \( km \) zeros are often appended at the end of an information sequence to clear the shift register contents.
  - Hence, \( kL \) message bits will produce \( n(L+m) \) output bits.
  - The **effective code rate** is therefore given by:
    \[
    \hat{R} = \frac{kL}{n(L+m)}
    \]
  - Since \( L \) is often much larger than \( m \), \( \hat{R} \approx \frac{k}{n} \).
  - \( k/n \) is named the **code rate** of a convolutional code.

---

10.5 Convolutional codes

- **Polynomial expression of convolutional codes**
  - **Example 10.5** (Slide 10-63)
    - \( D \) is used instead of \( X \) because the flip-flop (i.e., one time-unit delay) is often denoted by \( D \).
    - \( g^{(1)}(D) = 1 + D + D^2 \)
    - \( g^{(2)}(D) = 1 + D^2 \)
    - \( m(D) = 1 + D^3 + D^4 \)
    - \( c^{(1)}(D) = g^{(1)}(D)m(D) = 1 + D + D^2 + D^3 + D^4 \)
    - \( c^{(2)}(D) = g^{(2)}(D)m(D) = 1 + D^2 + D^3 + D^4 + D^5 + D^6 \)
10.5 Convolutional codes

Buffer two bits at a time

\[ c^{(1)}(D) = D^6 + D^3 + D^2 + D + 1 \]

Input

\[ 11001 \]

\[ m(D) = D^4 + D^3 + 1 \]

\[ m = \{10011\} \]

Output

\[ c^{(2)}(D) = D^6 + D^5 + D^4 + D^3 + D^2 + 1 \]

\[ c = \{11, 10, 11, 11, 01, 01, 11\} \]

Graphical expressions of convolutional codes

- Code tree
  - \[ m = \{10011\} \]
  - \[ c = \{11, 10, 11, 11, 01, 01, 11\} \]

- Code trellis
  - \[ \{100m_3m_4 \ldots\} \]
  - \[ \{000m_3m_4 \ldots\} \]
  - generate the same “next code symbol”
10.5 Convolutional codes

- Code trellis (continue)

Solid line : 0
Dashed line : 1
Append two zeros to clear the shift-register contents

- State diagram
10.6 Maximum likelihood decoding of convolutional codes

- **Likelihood function** (i.e., probability function)
  \[ p(r|c), \text{ where } r = c + n. \]

- **Maximum-likelihood decoding**
  \[ \hat{c} = \max_{c \in C} p(r|c) \]

- For equal prior probability, ML decoding minimizes the error rate.

---

10.6 Maximum likelihood decoding of convolutional codes

- **Minimum distance decoding**  
  - For an additive noise,
    \[ \hat{c} = \max_{c \in C} p(r|c) \]
    \[ = \max_{c \in C} q(r - c), \text{ where } q \text{ is the distribution of } n \]
    \[ = \min_{c \in C} d(r, c) \]
  
  if \( q(r - c) \) is a monotonically decreasing function of \( d(r, c) \).
10.6 Maximum likelihood decoding of convolutional codes

- Example (of distance function): AWGN \( r = c + n \)

\[
q(n) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left\{-\frac{||n||^2}{2\sigma^2}\right\}
\]

\[
d(r, c) = ||r - c||^2
\]

- Example (of distance function): BSC \( r = c \oplus n \)

\[
q(n) = p^{w_H(n)}(1 - p)^{N - w_H(n)}
\]

\[
d(r, c) = w_H(r \oplus c)
\]

---

10.6 Maximum likelihood decoding of convolutional codes

- Viterbi algorithm (for minimum distance decoding over code trellis)

- Optimality

![Trellis diagram with survivor paths]

One survivor path (solid line) for each intermediate node
Assume that the all-zero sequence is transmitted.

\[ d(r, c) = w_H(r \oplus c) \]

It is possible that the ML codeword or the minimum distance codeword is not equal to the transmitted codeword.
10.6 Maximum likelihood decoding of convolutional codes

- Free distance of convolutional codes
  - Under binary symmetric channels (BSCs), a convolutional code with free distance $d_{\text{free}}$ can correct $t$ errors iff $d_{\text{free}}$ is greater than $2t$.

- Question: How to determine $d_{\text{free}}$?
  - Answer: By signal-flow graph.
Exponent of $D$ = Hamming weight on the branch
Exponent of $L$ = Length of the branch

State diagram

Signal graph

Example:
- Input 100 generates a codeword of length 3 (branches) with weight 5.
- Input 10100 generates a codeword of length 5 (branches) with weight 6.

A zero-weight input generates a zero-weight code pattern.

\[
\begin{align*}
b &= D^2 L \cdot a_0 + L \cdot c \\
c &= DL \cdot b + DL \cdot d \\
d &= DL \cdot b + DL \cdot d \\
a_1 &= D^2 L \cdot c
\end{align*}
\]

\[
\frac{a_1}{a_0} = \frac{D^5 L^3}{1 - DL(1 + L)} \text{ (See the next slide)}
\]

\[
\frac{1}{1 - x} = 1 + x + x^2 + x^3 + \cdots
\]

\[
\Rightarrow \frac{a_1}{a_0} = D^5 L^3 \sum_{i=0}^{\infty} [DL(1 + L)]^i
\]

\[
= D^5 L^3 + D^6(L^4 + L^5) + D^7(L^5 + 2L^6 + L^7) + \cdots
\]

100(a_0bca_1)(L^3) → codeword of weight 3

1100(a_0bdca_1)(L^4) → codeword of weight 6

10100(a_0bebca_1)(L^5) → codeword of weight 6
10.6 Maximum likelihood decoding of convolutional codes

- Since the distance transfer function $T(D, 1) = D^5 + 2D^6 + 4D^7 + \ldots$ enumerates the number of codewords that are a given distance apart, it follows that $d_{free} = 5$ in the previous example.

- A convolutional code may be subject to catastrophic error propagation.
  - **Catastrophic error propagation** = A finite number of transmission errors may cause infinite number of decoding errors.
  - A code with potential catastrophic error propagation is named a catastrophic code.
A non-zero-weight input generates a zero-weight code pattern.

\[ g^{(1)}(D) = D + D^2 \]

\[ g^{(2)}(D) = 1 + D \]

\[ \mathbf{m} = (111111 \ldots) \]

\[ m(D) = 1 + D + D^2 + D^3 + \ldots \]

\[ \Rightarrow c^{(1)}(D) = g^{(1)}(D)m(D) = E \]

\[ \Rightarrow c^{(2)}(D) = 1 \]

\[ c = (01, 10, 00, 00, 00, 00, 00, \ldots) \]

The maximum-likelihood decoder will decode \( \mathbf{r} = (00, 00, 00, 00, 00, 00, 00, \ldots) \) as the all-zero codeword corresponding to \( \mathbf{m} = (000000 \ldots) \) even if there are only two transmission errors!
Alternative definition of catastrophic codes: A code for which an infinite weight input causes a finite weight output
- In terms of the state diagram, a catastrophic code will have a loop corresponding to a nonzero input for which all the output bits are zeros.
- It can be proved that a systematic convolutional code cannot be catastrophic.
- Unfortunately, the free distance of systematic codes is usually smaller than that of nonsystematic codes.

<table>
<thead>
<tr>
<th>Constraint length</th>
<th>Systematic</th>
<th>Nonsystematic</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>10</td>
</tr>
</tbody>
</table>

Maximum free distances attainable for convolutional codes of rate 1/2.

10.6 Maximum likelihood decoding of convolutional codes
- Asymptotic coding gain
- Hard decision decoding
- Section 6.3 (cf. Slide 6-32) has established that for BPSK transmission, the hard-decision error for each code bit is given by:
  \[ p = P(\text{Error}) = \Phi \left( -\sqrt{\frac{E}{N_0}} \right) \]
  - The error of convolutional codes (particularly at high SNR) is dominated by the “two codewords” whose pairwise Hamming distance is equal to \( d_{\text{free}} \).
  - Thus, the (code)word error rate (WER) can be analyzed via an equivalent binary symmetric channel (BSC) with crossover probability \( p \).
Equivalently \( x_j = s_{j,m} \oplus w_j \) for \( j = 1, \ldots, d_{\text{free}} \)
where \( s_{j,0} = 0, s_{j,1} = 1, \) and \( x_j, w_j \in \{0, 1\} \)

\[ \hat{m} = \arg \max \left\{ P(\mathbf{x} | \mathbf{s}_0), P(\mathbf{x} | \mathbf{s}_1) \right\} \]

\[ \hat{m} = \arg \max \left\{ p^{w_{H}(\mathbf{x})} (1 - p)^{d_{\text{free}} - w_{H}(\mathbf{x})}, (1 - p)^{w_{H}(\mathbf{x})} p^{d_{\text{free}} - w_{H}(\mathbf{x})} \right\} \]

\[ w_{H}(\mathbf{x}) \leq \frac{d_{\text{free}}}{s_1}, \quad \text{if } p < \frac{1}{2} \]

Dominant pairwise error

\[ P(\mathbf{s}_0 \text{ transmitted}) P \left( w_{H}(\mathbf{x}) > \frac{d_{\text{free}}}{2} \mid \mathbf{s}_0 \text{ transmitted} \right) \]

\[ + P(\mathbf{s}_1 \text{ transmitted}) P \left( w_{H}(\mathbf{x}) < \frac{d_{\text{free}}}{2} \mid \mathbf{s}_1 \text{ transmitted} \right) \]

\( \leq \left[ 4p(1 - p) \right]^{d_{\text{free}}/2} \leq (4p)^{d_{\text{free}}/2} \]

\[ \leq \exp \left\{ -d_{\text{free}} E/(2N_0) \right\} = \exp \left\{ -d_{\text{free}} RE_0/(2N_0) \right\} \]

\( 4\Phi(-x) = 2\text{erfc} \left( \frac{x}{\sqrt{2}} \right) \leq \frac{4}{x\sqrt{2\pi}} e^{-x^2/2} \leq e^{-x^2/2} \text{ for } x > 4\sqrt{2\pi} \)

For your information, assuming \( d_{\text{free}} \) is odd, we derive

\[ \Rightarrow \text{Dominant pairwise error} \]

\[ = P(\mathbf{s}_0 \text{ transmitted}) P \left( w_{H}(\mathbf{x}) > \frac{d_{\text{free}}}{2} \mid \mathbf{s}_0 \text{ transmitted} \right) \]

\[ + P(\mathbf{s}_1 \text{ transmitted}) P \left( w_{H}(\mathbf{x}) < \frac{d_{\text{free}}}{2} \mid \mathbf{s}_1 \text{ transmitted} \right) \]

\[ = \Pr \left[ W_1 + W_2 + \ldots + W_{d_{\text{free}}} > \frac{d_{\text{free}}}{2} \right], \]

where \( \{W_j\} \) i.i.d. with \( P(W_j = 1) = 1 - P(W_j = 0) = p \)

\[ = \Pr \left[ e^{\theta(W_1 + W_2 + \ldots + W_{d_{\text{free}}})} > e^{d_{\text{free}}/2} \right], \quad \text{where } e^\theta = (1 - p)/p \]

\[ \leq \left( \frac{E[e^{\theta W_1}]}{e^{d_{\text{free}}/2}} \right)^{d_{\text{free}}} = \left( \frac{pe^\theta + 1 - p}{e^{d_{\text{free}}/2}} \right)^{d_{\text{free}}} \]

\[ = [4p(1 - p)]^{d_{\text{free}}/2} \]

\[ \square \]
- Soft decision decoding (can be analyzed via an equivalent binary-input additive white Gaussian noise channel)
  - WER of convolutional codes (particularly at high SNR) is dominated by the “two codewords” whose pairwise Hamming distance is equal to \( d_{\text{free}} \).

\[ \Rightarrow \text{Equivalently } x_j = s_{j,m} + w_j \text{ for } j = 1, \ldots, d_{\text{free}} \]

where \( s_{j,0} = -\sqrt{E} \) and \( s_{j,1} = \sqrt{E} \)

\[ \Rightarrow \hat{m} = \arg \max \left\{ P(\mathbf{x} | s_0), P(\mathbf{x} | s_1) \right\} \]

\[ \Rightarrow \hat{m} = \arg \max \left\{ \prod_{j=1}^{d_{\text{free}}} e^{-(x_j + \sqrt{E})^2/2\sigma^2}, \prod_{j=1}^{d_{\text{free}}} e^{-(x_j - \sqrt{E})^2/2\sigma^2} \right\} \]

\[ \Rightarrow x = \sum_{j=1}^{d_{\text{free}}} \begin{cases} s_0 & \text{if } x_j \leq 0 \\ s_1 & \text{if } x_j > 0 \end{cases} \]

\[ P(-d_{\text{free}} \sqrt{E}, d_{\text{free}} \sigma^2) \quad \mathcal{N}(0, \sigma^2) \]

\[ r^2 = N_0/2 \text{ is the variance of } w \]

- Based on the decision rule \( x = \sum_{j=1}^{d_{\text{free}}} x_j \) \( \leq 0 \)

Dominant pairwise error

\[ = P(s_0 \text{ transmitted}) P(x > 0 | s_0 \text{ transmitted}) + P(s_1 \text{ transmitted}) P(x < 0 | s_1 \text{ transmitted}) \]

\[ = \frac{1}{2} \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi d_{\text{free}} \sigma^2}} e^{-(x-d_{\text{free}} \sqrt{E})^2/2d_{\text{free}} \sigma^2} dx \]

\[ + \frac{1}{2} \int_0^{+\infty} \frac{1}{\sqrt{2\pi d_{\text{free}} \sigma^2}} e^{-(x+d_{\text{free}} \sqrt{E})^2/2d_{\text{free}} \sigma^2} dx \]

\[ = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi d_{\text{free}} \sigma^2}} e^{-(x-d_{\text{free}} \sqrt{E})^2/2d_{\text{free}} \sigma^2} dx \]

\[ = \Phi \left( \frac{0 - d_{\text{free}} \sqrt{E}}{\sqrt{d_{\text{free}} \sigma^2}} \right) = \Phi \left( \frac{\sqrt{E}}{\sqrt{N_0}} \right) \]

\[ = \Phi \left( \frac{-d_{\text{free}} R E_b}{N_0} \right) \leq \exp \left\{ -d_{\text{free}} R E_b / N_0 \right\} \]

\[ \Phi(x) = \frac{1}{2} \text{erfc} \left( \frac{x}{\sqrt{2}} \right) \leq \frac{1}{x \sqrt{2\pi}} e^{-x^2/2} \leq e^{-x^2/2} \text{ for } x > 1/\sqrt{2\pi} \]
Asymptotic coding gain (here, asymptotic = at high SNR) $G_a$

- The performance gain due to coding (i.e., the performance gain of a coded system against an uncoded system)

\[
\text{Uncoded BPSK } \Phi \left(-\sqrt{\frac{2E_b}{N_0}}\right) \leq \exp \left\{ -\frac{E_b}{N_0} \right\}
\]

Coded system \[\exp \left\{ -G_a \frac{E_b}{N_0} \right\}\]

Convolutional coded BPSK with hard-decision decoding \[\exp \left\{ -\left(\frac{d_{\text{free}}R}{2}\right) \frac{E_b}{N_0} \right\}\]

Convolutional coded BPSK with soft-decision decoding \[\exp \left\{ -d_{\text{free}}R \frac{E_b}{N_0} \right\}\]

Asymptotic coding gain (at high SNR)

\[
\text{Convolutional coded BPSK with hard-decision decoding } G_a = \frac{d_{\text{free}}R}{2} = 10 \log_{10} \left( \frac{d_{\text{free}}R}{2} \right) \text{ dB}
\]

\[
\text{Convolutional coded BPSK with soft-decision decoding } G_a = d_{\text{free}}R = 10 \log_{10}(d_{\text{free}}R) \text{ dB}
\]

Asymptotic coding gain = $\frac{E_b}{N_0}$ uncoded $\frac{E_b}{N_0}$ coded under the same error rate

\[\exp \left\{ -\left(\frac{E_b}{N_0}\right) \text{ uncoded} \right\} \approx P_e \approx \exp \left\{ -G_a \left(\frac{E_b}{N_0}\right) \text{ coded} \right\}\]
Asymptotic coding gain (at high SNR)

\[ E_b/N_0 \text{ dB} \quad P_e \quad G_a \]

Error (log-scale)

10.7 Trellis-coded modulation

In the previous section, encoding is performed separately from modulation in the transmitter, and likewise for decoding and detection in the receiver.

To attain more effective utilization of the available bandwidth and power, coding and modulation have to be treated as a single entity, e.g., trellis-coded modulation.

- Instead of selecting codewords from “code bit domain”, we choose codewords from “signal constellation domain”.

© Po-Ning Chen@cm.nctu

Chapter 10-93

Chapter 10-94
Partitioning of 8-PSK constellation that shows $d_0 < d_1 < d_2$.

Partitioning of 16-QAM constellation that shows $d_0 < d_1 < d_2 < d_3$. 
10.7 Trellis-coded modulation

- Codeword versus code signal

0000
0011
1100
1111

Select 4 out of 16 possibilities
(The bit patterns are dependent temporally so that these bit patterns exhibit “error correcting capability”.)

\[
\begin{array}{cc}
0 & \pi \\
\frac{\pi}{2} & \frac{3\pi}{2} \\
\pi & 0 \\
\frac{3\pi}{2} & \frac{\pi}{2}
\end{array}
\]

Select 4 out of 16 possibilities from QPSK constellation
(The signal patterns are dependent temporally so these signal patterns exhibit “error correcting capability”.)

- Trellis codeword versus trellis code signal

- The next code bit is a function of the current trellis state and some number of the previous information bits.

- The next code signal is a function of the current trellis state and some number of the previous information signals.
Example of trellis-coded modulation
- 4-state Ungerboeck 8-PSK code
  - Code rate = $2$ bits/symbol