1. (10%) The capacity for the **continuous-input** AWGN channel is given by

$$C = \frac{1}{2}\log_2\left(1+\frac{P}{\sigma^2}\right) \quad \text{bits per channel usage,}$$

where $\sigma^2 = \frac{N_0}{2}$, and P (which sets the power constraint on the system, i.e., $\frac{1}{n}\sum_{i=1}^{n} E[|X_i|^2] \le P$) is measured in **Joule per channel usage**. What is the minimum E_b/N_0 required for reliable transmission, subject to $R = k/n = \frac{1}{4}$ bits/channel usage?

Solution. The energy of n transmissions (equivalently, n channel usages) is kE_b. Thus, in average, we have

$$P = \frac{kE_b}{n} = RE_b \text{ Joule/channel usage.}$$

The noise power experienced in each transmission is $\sigma^2 = \frac{N_0}{2}$. Shannon then said that reliable transmission is possible only when

$$R \text{ bits/channel usage} < C \text{ bits/channel usage.}$$

By Shannon's formula, we know reliable transmission is possible if

$$R = \frac{k}{n} = \frac{1}{4} \text{ bits/channel usage} < \frac{1}{2}\log_2\left(1+\frac{RE_b}{\frac{N_0}{2}}\right) = \frac{1}{2}\log_2\left(1+\frac{E_b}{2N_0}\right) \text{ bits/channel usage,}$$

which is equivalent to

$$2(\sqrt{2}-1) < \frac{E_b}{N_0}.$$

Hence, the minimum E_b/N_0 required for reliable transmission over the continuous-input AWGN channel, subject to $R = k/n = \frac{1}{4}$ bits/channel usage, is $2(\sqrt{2}-1)$ (≈ -0.817 dB).

2. Give a generator matrix:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix}$$

(a) (6%) Is it a systematic code? Justify how you get the answer.

(b) (8%) List all codewords of this code.

(c) (6%) From (c), check if it is a cyclic code. If affirmative, give the generator polynomial of this code.
Hint: $c(X) = a(X)g(X)$

(d) (6%) Find the syndrome (or syndrome polynomial) if the received vector is 0111001.

Solution.

(a) Since the generator matrix contains an identity matrix, i.e.,

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

it is a systematic code.

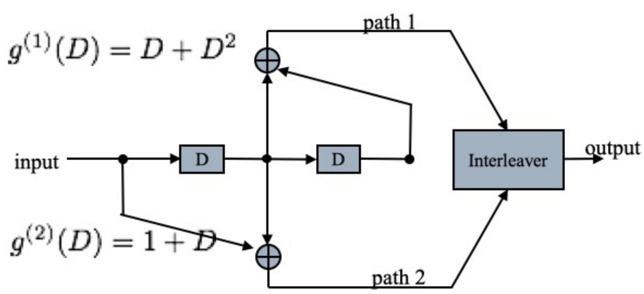
(b)

$$\underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}}_{\text{all 8 possible inputs}} \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

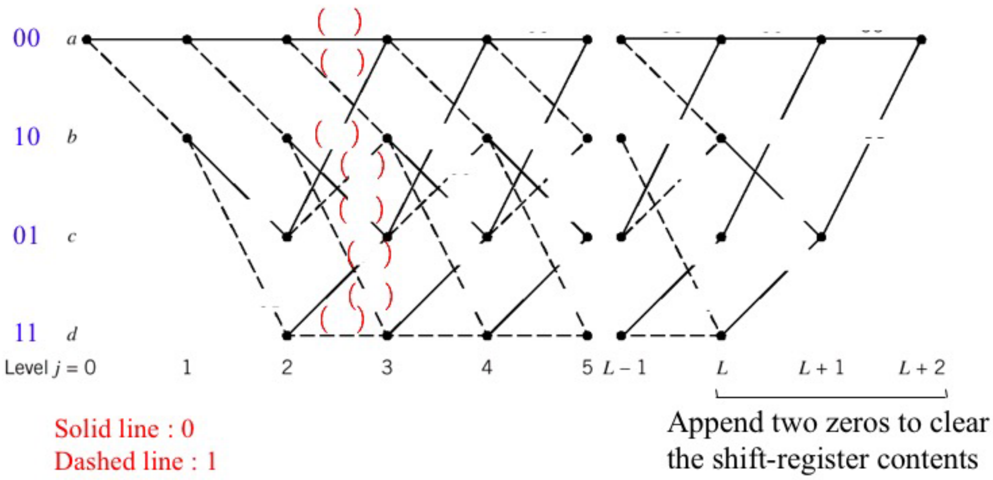
(c) Since from (c), we observe that all cyclic shifts of a codeword are codewords, it is a cyclic code. The generator polynomial $g(X)$ can be obtained via setting $a(X) = 1$, i.e., $a_0a_1a_2 = 100$. We then obtain from the codeword 1011100 corresponding to input 100 that $g(X) = c(X) = 1 + X^2 + X^3 + X^4$.

(d) Since 0111001 is a codeword, the syndrome is 0000.

3. (a) (8%) For the convolutional code given in the following,



fill in the code bits inside the parentheses on the code trellis below. Please give your answer by providing the code bits associated with the branch, for example, "the code bits from state a to state a are xx."



(b) (8%) Draw the state diagram of this convolutional code.

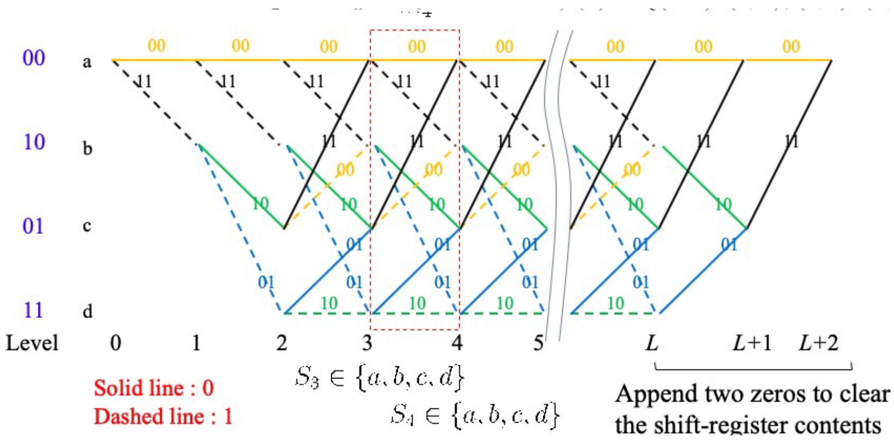
Solution.

(a) Let me give you an example of how these parentheses can be filled. At state c, where the two shift registers contain 01, when input is 0, then path 1 gives $0 \oplus 1 = 1$ and path 2 gives $0 \oplus 0 = 0$; thus, the output code bits are 10 (the next state is 00). When input is 1, then path 1 gives $0 \oplus 1 = 1$ and path 2 gives $1 \oplus 0 = 1$; thus, the output code bits are 11 (and the next state is 10).

For the solution, see Slide IDC 7-84.

(b) See Slides IDC 7-84.

4. (a) (6%) In the dotted box below, what is the set of transitions $\mathcal{B}_{3,4}(1) \subset \mathcal{S}_3 \times \mathcal{S}_4$ corresponding to symbol 1?



(b) (6%) The BCJR algorithm intends to minimize a particular bit error such as the one from level 3 to level 4. The decision is based on the log-likelihood ratio

$$l(4) = \log \frac{\sum_{(S_3,S_4)\in\mathcal{B}_{3,4}(1)} P(S_3, S_4, \mathbf{r})}{\sum_{(S_3,S_4)\in\mathcal{B}_{3,4}(0)} P(S_3, S_4, \mathbf{r})}$$

The technique is to decompose $P(S_3, S_4, \mathbf{r})$ into the product of $\alpha(S_3)$, $\beta(S_4)$ and $\gamma(S_3, S_4)$, where $\alpha(S_3)$ is only a function of the portion of the received vector before level 3, $\beta(S_4)$ is only a function of the portion of the received vector after level 4, and $\gamma(S_3, S_4)$ is only a function of the portion of the received vector between levels 3 and 4. Please complete the following derivation.

$$\begin{aligned}
P(S_3, S_4, r_1^N) &= P(S_3, S_4, \underbrace{r_1^6}_{\text{past}}, \underbrace{r_7^8}_{\text{now}}, \underbrace{r_9^N}_{\text{future}}) \\
&= P(r_9^N | \cdot, \cdot, \cdot, \cdot) P(\cdot, \cdot, \cdot, \cdot) \\
&= P(r_9^N | \cdot) P(\cdot, \cdot, \cdot, \cdot) \\
&\quad \text{(Explain why we can simplify the first term.)} \\
&= P(r_9^N | \cdot) P(\cdot, \cdot) P(S_4, r_7^8 | \cdot, \cdot) \\
&= P(r_9^N | \cdot) P(\cdot, \cdot) P(S_4, r_7^8 | \cdot) \\
&\quad \text{(Explain why we can simplify the last term.)}
\end{aligned}$$

(c) (10%) The computation of α and β functions can be done recursively, provided that

γ function is given. Please complete the recursive derivation below.

$$\begin{aligned}
\alpha(S_3) &= P(S_3, r_1^6) \\
&= \sum_{S_2 \in \{a,b,c,d\}} P(S_2, S_3, r_1^4, r_5^6) \\
&= \sum_{S_2 \in \{a,b,c,d\}} P(S_2, \cdot) P(S_3, \cdot | S_2, \cdot) \\
&= \sum_{S_2 \in \{a,b,c,d\}} P(S_2, \cdot) P(S_3, \cdot | S_2) \\
&= \sum_{S_2 \in \{a,b,c,d\}} \alpha(S_2)\gamma(S_2, S_3)
\end{aligned}$$

and

$$\begin{aligned}
\beta(S_4) &= P(r_9^N | S_4) \\
&= \sum_{S_5 \in \{a,b,c,d\}} P(S_5, r_9^{10}, r_{11}^N | S_4) \\
&= \sum_{S_5 \in \{a,b,c,d\}} P(\cdot | S_4, S_5, \cdot) P(S_5, \cdot | S_4) \\
&= \sum_{S_5 \in \{a,b,c,d\}} P(\cdot | S_5) P(S_5, \cdot | S_4) \\
&= \sum_{S_5 \in \{a,b,c,d\}} \beta(S_5)\gamma(S_4, S_5)
\end{aligned}$$

(d) (6%) The decomposition of γ-function into the product of "systematic" part, "parity" part and "prior" part can be done as follows.

$$\begin{aligned}
\gamma(S_3, S_4) &= P(S_4, r_7^8 | S_3) \\
&= P(r_7^8 | x_7^8(S_3, S_4)) P(S_4 | S_3) \\
&= P(r_7^8 | m_4, x_8(S_3, S_4)) P(S_4 | S_3) \\
&= \underbrace{P(r_7|m_4)}_{\text{part 1}} \underbrace{P(r_8|x_8(S_3, S_4))}_{\text{part 2}} \underbrace{P(S_4|S_3)}_{\text{part 3}}
\end{aligned}$$

Which one of the three parts corresponds to the prior probability $P(m_4 = 1)$ and $P(m_4 = 0)$?

Solution.

(a) $\mathcal{B}_{3,4}(1) = \{(a, b), (b, d), (c, b), (d, d)\}$

(b)

$$\begin{aligned}
P(S_3, S_4, r_1^N) &= P(S_3, S_4, \underbrace{r_1^6}_{\text{past}}, \underbrace{r_7^8}_{\text{now}}, \underbrace{r_9^N}_{\text{future}}) \\
&= P(r_9^N | S_3, S_4, r_1^6, r_7^8) P(S_3, S_4, r_1^6, r_7^8) \\
&= P(r_9^N | S_4) P(S_3, S_4, r_1^6, r_7^8) \\
&\quad \text{(Because } (S_3, r_1^6, r_7^8) \to S_4 \to r_9^N \text{ forms a Markov chain.)} \\
&= P(r_9^N | S_4) P(S_3, r_1^6) P(S_4, r_7^8 | S_3, r_1^6) \\
&= P(r_9^N | S_4) P(S_3, r_1^6) P(S_4, r_7^8 | S_3) \\
&\quad \text{(Because } r_1^6 \to S_3 \to (S_4, r_7^8) \text{ forms a Markov chain.)}
\end{aligned}$$

(c) See Slides IDC 8-32 and IDC 8-33.

(d) Part 3, i.e., $P(S_4|S_3)$.

5. (a) (6%) For a given parity-check matrix

$$H_{(n-k)\times n} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}_{4\times 6}$$

draw Forney's factor graph (or bipartite graph) with six variable nodes and four check nodes.

(b) (8%) The Gallager Sum-Product Algorithm can be described as follows.

Step 1. Initialization. Set $P_{i,j}^1 = p_j^1 = P(c_j = 1 | r_j)$ for $0 \le i \le 3$.

Step 2. Check Node Update (Horizontal Step). Perform

$$Q_{i,j}^1 = \frac{1}{2}\left(1 + \prod_{k \in \text{Check}(i) \setminus \{j\}} (2P_{i,k}^1 - 1)\right)$$

where Check(i) is the set of indices of variable nodes that connect to check node i.

Step 3. Variable Node Update (Vertical Step). Perform

$$P_{i,j}^1 = \frac{p_j^1 \prod_{k \in \text{Bit}(j) \setminus \{i\}} Q_{k,j}^1}{p_j^1 \prod_{k \in \text{Bit}(j) \setminus \{i\}} Q_{k,j}^1 + (1 - p_j^1) \prod_{k \in \text{Bit}(j) \setminus \{i\}} (1 - Q_{k,j}^1)}$$

where Bit(j) is the set of indices of check nodes that connect to variable node j.

Step 4. Decision. Compute

$$P_j^1 = \frac{p_j^1 \prod_{k \in \text{Bit}(j)} Q_{k,j}^1}{p_j^1 \prod_{k \in \text{Bit}(j)} Q_{k,j}^1 + (1 - p_j^1) \prod_{k \in \text{Bit}(j)} (1 - Q_{k,j}^1)}$$

and

$$\hat{c}_j = \begin{cases} 1, & \text{if } P_j^1 > \frac{1}{2} \\ 0, & \text{if } P_j^1 < \frac{1}{2} \end{cases}$$

Step 5. Termination. If $\hat{\boldsymbol{c}}H^T = \boldsymbol{0}$, the algorithm stops;
else set $p_j^1 = P_j^1$ and go to Step 2.

Suppose the all-zero codeword is transmitted (where we map $\{0, 1\}$ to $\{1, -1\}$) and initially, we have

$$p_j^1 = P((-1)^{c_j} = 1|r_j) = 0.6 \text{ for } 1 \le j \le 5$$

and

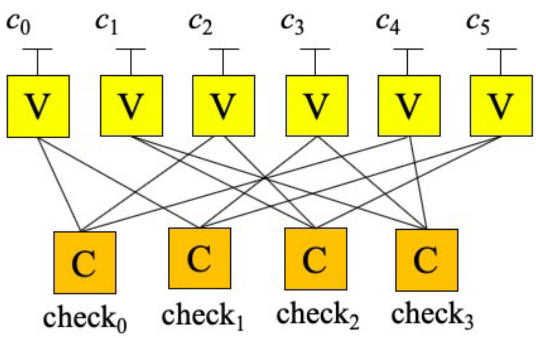
$$p_0^1 = P((-1)^{c_0} = 1|r_0) = 0.1.$$

What are the values of $Q_{0,0}^1$ and $Q_{1,0}^1$?

(c) (6%) Perform the decision step to find P_0^1 using the result in (b).

Solution.

(a)



(b) Initially, we have

$$\begin{bmatrix} P_{0,0}^1 & P_{1,0}^1 & P_{2,0}^1 & P_{3,0}^1 \\ P_{0,1}^1 & P_{1,1}^1 & P_{2,1}^1 & P_{3,1}^1 \\ P_{0,2}^1 & P_{1,2}^1 & P_{2,2}^1 & P_{3,2}^1 \\ P_{0,3}^1 & P_{1,3}^1 & P_{2,3}^1 & P_{3,3}^1 \\ P_{0,4}^1 & P_{1,4}^1 & P_{2,4}^1 & P_{3,4}^1 \\ P_{0,5}^1 & P_{1,5}^1 & P_{2,5}^1 & P_{3,5}^1 \end{bmatrix} = \begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 \\ 0.6 & 0.6 & 0.6 & 0.6 \\ 0.6 & 0.6 & 0.6 & 0.6 \\ 0.6 & 0.6 & 0.6 & 0.6 \\ 0.6 & 0.6 & 0.6 & 0.6 \\ 0.6 & 0.6 & 0.6 & 0.6 \end{bmatrix}$$

Also, we know Check(0)=$\{0, 2, 4\}$, Check(1)=$\{0, 3, 5\}$, Check(2)=$\{1, 2, 5\}$ and Check(3)=$\{1, 3, 4\}$. This gives

$$Q_{0,0}^1 = \frac{1}{2}\left(1 + \prod_{k \in \text{Check}(0) \setminus \{0\}} (2P_{0,k}^1 - 1)\right) = \frac{1}{2}\left(1 + \prod_{k \in \{2,4\}} (2P_{0,k}^1 - 1)\right) = 0.52$$

and

$$Q_{1,0}^1 = \frac{1}{2}\left(1 + \prod_{k \in \text{Check}(1) \setminus \{0\}} (2P_{1,k}^1 - 1)\right) = 0.52$$

(c) With Bit(0)=$\{0,1\}$, we have

$$
\begin{aligned}
P_0^1 &= \frac{p_0^1 \prod_{k\in\mathrm{Bit}(0)} Q_{k,0}^1}{p_0^1 \prod_{k\in\mathrm{Bit}(0)} Q_{k,0}^1 + (1-p_0^1) \prod_{k\in\mathrm{Bit}(0)} (1-Q_{k,0}^1)} \\
&= \frac{0.1 \cdot 0.52 \cdot 0.52}{0.1 \cdot 0.52 \cdot 0.52 + 0.9 \cdot 0.48 \cdot 0.48} \\
&\approx 0.1154
\end{aligned}
$$