# Solution Sheet 4

Last modified November 18, 2019

## Problem 1

1. The easiest way to verify that a code is a linear block code is to come up with a generator matrix and show that it generates the 8 codewords. An example is

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

   consisting of codewords 5, 3, and 2. The code parameters are $n = 7$, $k = 3$ (since there are $2^3 = 8$ codewords), and $d_{\mathsf{min}} = 4$ (since the minimum distance is also the minimum Hamming weight).

2. We already have a generator matrix of the form $\boldsymbol{G}_{\mathsf{s}} = [\boldsymbol{P}_{3\times 4}\, \boldsymbol{I}_3]$,

$$\boldsymbol{G}_{\mathsf{s}} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

   A corresponding 4 by 7 parity check matrix is

$$\boldsymbol{H}_{\mathsf{s}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

   You can verify that $\boldsymbol{H}_{\mathsf{s}}\boldsymbol{G}_{\mathsf{s}}^{\mathsf{T}} = \mathbf{0}$.

3. Over the BSC channel, the code can correct any error $\boldsymbol{e}$ with

$$\begin{aligned} w_{\mathsf{H}}(\boldsymbol{e}) &\leq \left\lfloor \frac{d_{\mathsf{min}} - 1}{2} \right\rfloor \\ &= 1. \end{aligned}$$

   The same code can also detect any error $\boldsymbol{e}$ with

$$\begin{aligned} w_{\mathsf{H}}(\boldsymbol{e}) &< d_{\mathsf{min}} \\ &= 4. \end{aligned}$$

4. The syndrome table contains $2^4 - 1 = 15$ entries. We first compute the syndromes of all errors of weight 1. These are the columns of the parity check matrix, since for an $\boldsymbol{e}$ with exactly a single one, $\boldsymbol{H}_s\boldsymbol{e}^{\mathsf{T}}$ is a column on $\boldsymbol{H}_s$. This gives us a partial syndrome table, show in Table 1. The remaining entries can be found by considering the equation

$$\mathbf{s} = \boldsymbol{H}_s\boldsymbol{e}^{\mathsf{T}}$$

   so that

$$\begin{aligned} s_1 &= e_1 + e_6 + e_7 \\ s_2 &= e_2 + e_5 + e_7 \\ s_3 &= e_3 + e_5 + e_6 \\ s_4 &= e_4 + e_5 + e_6 + e_7 \end{aligned}$$

| error pattern | syndrome |
|---|---|
| 0001000 | 0001 |
| 0010000 | 0010 |
| | 0011 |
| 0100000 | 0100 |
| | 0101 |
| | 0110 |
| 0000100 | 0111 |
| 1000000 | 1000 |
| | 1001 |
| | 1010 |
| 0000010 | 1011 |
| | 1100 |
| 0000001 | 1101 |
| | 1110 |
| | 1111 |

*Table 1:* Partial syndrome table

and thus

$$\begin{aligned} e_1 &= s_1 + e_6 + e_7 \\ e_2 &= s_2 + e_5 + e_7 \\ e_3 &= s_3 + e_5 + e_6 \\ e_4 &= s_4 + e_5 + e_6 + e_7. \end{aligned}$$

We see that for a fixed $\mathbf{s}$, we can vary $e_5, e_6, e_7$, which in term determines $e_1, e_2, e_3, e_4$. For a fixed $\mathbf{s}$, we need to find the combination of $e_5, e_6, e_7$ that leads to an $\boldsymbol{e}$ with the *smallest possible Hamming weight*. For instance, for $\mathbf{s} = [0011]$, we find that $e_1 = e_6 + e_7$, $e_2 = e_5 + e_7$, $e_3 = e_5 + e_6 + 1$ and $e_4 = e_5 + e_6 + e_7 + 1$. In this case, we find $\boldsymbol{e} = [0011000]$. Similarly, we can build up all the entries in the syndrome table. Note that some entries in the syndrome table are not unique.

5. When we receive [1111111], the syndrome is [1110], with corresponding error pattern [1110000]. Hence the output of the decoder is $\hat{\boldsymbol{c}} = [0001111]$. If your syndrome table was different, your decoder output could have also been any other codeword at Hamming distance 3 from $\bar{\boldsymbol{y}}$.

6. The decoder can correct all errors of weight one, and only some errors of weight 2. Hence, we expect our decoder to work well when $p < 1/7$. When $p > 1/7$, undetectable errors will occur (i.e., with two or more errors). When $p$ is very close to 1, the decoder should try to find a codeword at *maximal* Hamming distance to the observation, rather than minimum Hamming distance. The decoder could be implemented by creating a new observation $\tilde{\boldsymbol{y}} = \mathbf{1} + \bar{\boldsymbol{y}}$, where $\mathbf{1}$ is a vector of all ones and "+" is the binary addition. This new observation could then be used to compute the syndrome $\boldsymbol{s} = \boldsymbol{H}_{\mathrm{s}}\tilde{\boldsymbol{y}}^{\mathsf{T}}$, using the syndrome table from Table 2 to make a final decision.

| error pattern | syndrome |
|---|---|
| 0001000 | 0001 |
| 0010000 | 0010 |
| 0011000 | 0011 |
| 0100000 | 0100 |
| 0101000 | 0101 |
| 0110000 | 0110 |
| 0000100 | 0111 |
| 1000000 | 1000 |
| 1001000 | 1001 |
| 1010000 | 1010 |
| 0000010 | 1011 |
| 1100000 | 1100 |
| 0000001 | 1101 |
| 1110000 | 1110 |
| 1000100 | 1111 |

*Table 2:* Full syndrome table.

## Problem 2

1. $$\boldsymbol{G} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \qquad \Rightarrow \boldsymbol{P} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \qquad \boldsymbol{H} = \begin{pmatrix} \boldsymbol{P} & \boldsymbol{I}_{N-K} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix}$$

2. $d_{\mathsf{min}} = 2$, $R_{\mathsf{c}} = \frac{3}{4}$.

3. The code can correct 0 errors and can detect a single error or three errors, but not two or four.

4. We denote the probability of $j$ errors in a word consisting of $n$ bits by

$$f(j,n) = \binom{n}{j} p^j (1-p)^{n-j}.$$

An error is detected by the code if one or three errors occur in a word and undetected if two or four errors occur. The probability of detected and undetected error is therefore

$$P_{\mathrm{d}} = f(1,4) + f(3,4),$$
$$P_{\mathrm{ud}} = f(2,4) + f(4,4).$$

Considering our coded system setup, the probability of word error is

$$P_{w,\mathsf{c}} = P_{\mathrm{d}} \left(P_{\mathrm{d}} + P_{\mathrm{ud}}\right) + P_{\mathrm{ud}}.$$

For $p = 0.1$ we get

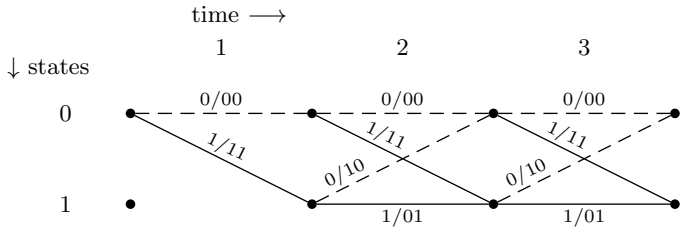
$$P_{w,\mathsf{c}} = 0.1502.$$

For the uncoded system the word error probability is

$$P_w = 1 - f\left(0,3\right) = 0.271.$$

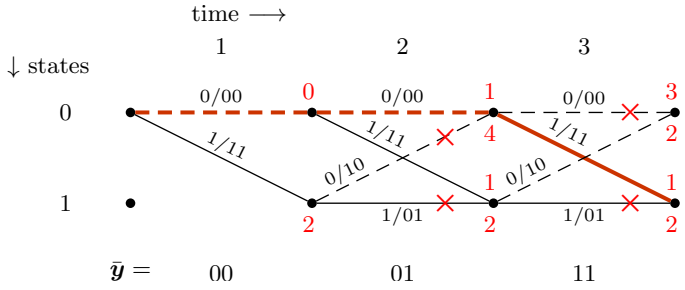
## Problem 3

1. The Trellis diagram looks as follows:




2. It can be seen that the minimum weight of a path emerging and reemerging to the all-zero path is 3, which is then the free distance of the code.

3. Executing the Viterbi algorithm for the given observation results in the following figure:




The final survivor path is marked in brown color and corresponds to the input (001) and output sequence (000011).

4. For such small information block sizes it is easily possible to perform a brute force approach to the decoding problem by simply listing all output sequences corresponding to all possible input sequences, together with the distance of the output sequence to the observation:

| input | output | distance |
|---|---|---|
| 000 | 000000 | 3 |
| 001 | 000011 | 1 |
| 010 | 001110 | 2 |
| 011 | 001101 | 2 |
| 100 | 111000 | 6 |
| 101 | 111011 | 4 |
| 110 | 110110 | 3 |
| 111 | 110101 | 3 |

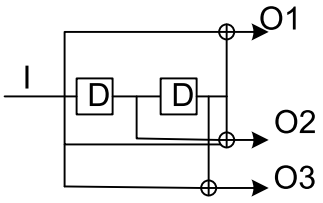
The brute-force approach thus confirms the obtained result.

5. The answer does not change at all, because, as long as $\epsilon < 0.5$, the optimal detector is a minimum distance detector, independent of $\epsilon$.

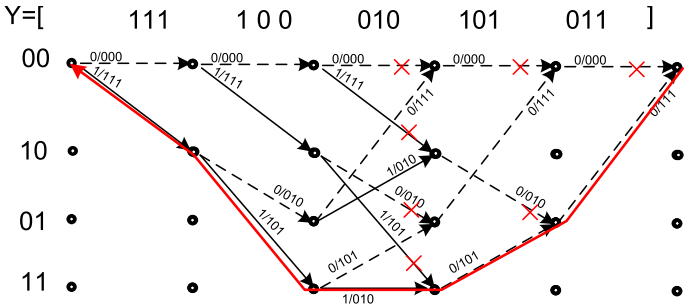
## Problem 4

1. As can be seen in the figure below, the constraint length is equal to 3. From encoder polynomials, it is clear that for 1 bit input encoders generates 3 bits output, hence the rate is equal $\frac{1}{3}$.

There are 15 received bits, which corresponds to 5 information bits. However, to reset the memories, 2 zeros bits must be inserted after the information bits. Therefore, the number of information bit is equal to 3.




2. The trellis diagram is shown in figure below. The red line is surviving path and then the decoded bit is:111




3. Considering Trellis diagram, if input bit be 11100, the encoded bit is $111, 101, 010, 101, 111$, hence two bits error occurred in channel.

## Problem 5

1. To find a generator matrix of the form $\boldsymbol{G}_{\text{s}} = [\boldsymbol{I}_3|\boldsymbol{P}]$, one can add the second to the third row and then the second and (new) third row to the first. A parity-check matrix then immediately follows from $\boldsymbol{H}_{\text{s}} = [\boldsymbol{P}^T|\boldsymbol{I}_4]$. In summary,

$$\boldsymbol{G}_{\text{s}} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix} \quad \text{and} \quad \boldsymbol{H}_{\text{s}} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

2. The codewords are obtained through $\boldsymbol{x} = \boldsymbol{u}\boldsymbol{G}$, where $\boldsymbol{u}$ are all possible 3-bit information words. Thus:

| $\boldsymbol{u}$ | $\boldsymbol{x}$ |
|---|---|
| 000 | 0000000 |
| 001 | 0011101 |
| 010 | 0100111 |
| 011 | 0111010 |
| 100 | 1001110 |
| 101 | 1010011 |
| 110 | 1101001 |
| 111 | 1110100 |

The weight spectrum is $A_0 = 1$, $A_4 = 7$, and $A_1 = A_2 = A_3 = A_5 = A_6 = A_7 = 0$.

3. The code parameters are $(7, 3, 4)$. The code rate is $R_c = 3/7$.

4. The code can *certainly* correct $\lfloor (d_{\text{H,min}} - 1)/2 \rfloor = 1$ error and detect up to (and including) 3 errors.

5. First, we compute the distance between $\bar{\boldsymbol{y}} = (1111111)$ and all codewords:

| $\boldsymbol{x}$ | $d_{\mathsf{H}}(\boldsymbol{x}, \bar{\boldsymbol{y}})$ |
|---|---|
| 0000000 | 7 |
| 0011101 | 3 |
| 0100111 | 3 |
| 0111010 | 3 |
| 1001110 | 3 |
| 1010011 | 3 |
| 1101001 | 3 |
| 1110100 | 3 |

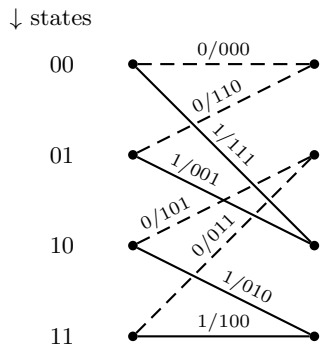
   Since for $\epsilon < 0.5$ the ML decoder is a minimum distance decoder, the ML codeword is any of the codewords except the all-zero codeword, so for example (1010011) and the corresponding information word is (101).

6. For $\epsilon > 0.5$, the ML decoder is a maximum distance decoder. The all-zero codeword is at maximum distance and hence the ML decoder decides for (0000000) with a corresponding information word of (000). Alternatively, one may flip all observation bits and still apply minimum distance decoding. This results in the same answer.

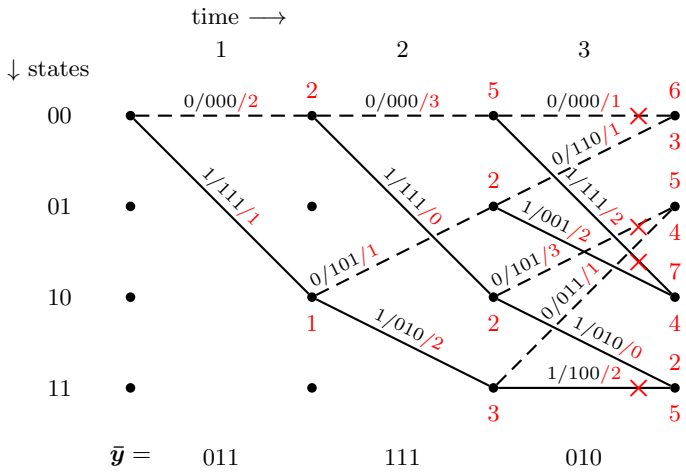
7. Error detection is achieved by multiplying the received word with the parity check matrix. In this case, we obtain $\boldsymbol{x}\boldsymbol{H}^{\mathsf{T}} = (1011)$. Since the syndrom is not the allzero vector, we know that an error occured during transmission. In general, if the error pattern $\boldsymbol{e}$ is itself a codeword, the error can certainly not be corrected. This is because in this case $\bar{\boldsymbol{y}} = \boldsymbol{x} + \boldsymbol{e}$ is itself a codeword, due to the linearity of the code. In this case we have $\bar{\boldsymbol{y}}\boldsymbol{H}^{\mathsf{T}} = (\boldsymbol{x} + \boldsymbol{e})\boldsymbol{H}^{\mathsf{T}} = \boldsymbol{x}\boldsymbol{H}^{\mathsf{T}} + \boldsymbol{e}\boldsymbol{H}^{\mathsf{T}} = \boldsymbol{0}$ and the error remains undetected.

### Problem 6

1. One trellis section should look like this:




2. In the figure below we show the complete trellis including all branch metrics, state metrics, and survivor paths:




Thus, we have $\hat{\boldsymbol{u}}_{\mathrm{ML}} = (011)$ and $\hat{\boldsymbol{c}}_{\mathrm{ML}} = (000111010)$.

3. For the brute-force approach, we enumerate all possible output sequence and compute the Hamming distance to the observation.

| input | output | distance to $\bar{\boldsymbol{y}}$ |
|---|---|---|
| 000 | 000 000 000 | 6 |
| 001 | 000 000 111 | 7 |
| 010 | 000 111 101 | 5 |
| 011 | 000 111 010 | 2 |
| 100 | 111 101 110 | 3 |
| 101 | 111 101 001 | 4 |
| 110 | 111 010 011 | 4 |
| 111 | 111 010 100 | 5 |

Again, we have $\hat{\boldsymbol{u}}_{\mathrm{ML}} = (011)$ and $\hat{\boldsymbol{c}}_{\mathrm{ML}} = (000111010)$.

4. We have three possible cases:
   - Ommitting $x_1$: free distance is 4 (minimal state transition $00 \to 10 \to 01 \to 00$)
   - Ommitting $x_2$: free distance is 4 (minimal state transition $00 \to 10 \to 11 \to 01 \to 00$)
   - Ommitting $x_3$: free distance is 5 (minimal state transition $00 \to 10 \to 01 \to 00$)

   Hence we should omit $x_3$.

## Problem 7

1. We remark that the operation $\boldsymbol{x}\boldsymbol{H}^T$ essentially adds together those columns in $\boldsymbol{H}$ where $\boldsymbol{x}$ has a one-entry. If we restrict $\boldsymbol{x}$ to have at most weight $d-1$ (that is, at most $d-1$ one-entries), then we know that $\boldsymbol{x}\boldsymbol{H}^T$ cannot add up to be $\boldsymbol{0}$. This is because otherwise $\boldsymbol{x}$ would be a codeword, and in particular a codeword with less weight than $d$, which contradicts the assumption that the code has minimum distance $d$.

2. Since the code has minimum distance $d$, there exists a codeword with $d$ one-entries, such that $\boldsymbol{x}\boldsymbol{H}^T = \boldsymbol{0}$. This is equivalent to saying that there exists at least one set of $d$ columns of $\boldsymbol{H}$ that are linearly dependent.