

BER and the Hamming Codes

1 MAP and ML Decision Rules

Throughout these notes, we shall stick entirely to *binary linear block codes*. Thus we shall be dealing with vector spaces over the binary field \mathbb{F} .

The following is deceptively simple.

DEFINITION 1.1. By a **length n , k bit (binary) linear block code** we mean a k -dimensional subspace $C \subseteq V$, where V is an n -dimensional vector space over \mathbb{F} . Sometimes we shall refer to C as an (n, k) linear block code.

The idea behind codes is that we wish to transmit k -bit messages across a noisy channel; to do this with some enhanced reliability, we build some measure of redundancy into the code. Thus the greater n exceeds k , the greater is the degree of redundancy. Of course, this redundancy is at the expense of **efficiency** of the code, which is defined as the ratio

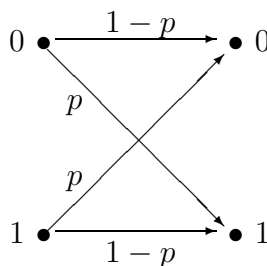
$$\text{Eff}(C) = \frac{\dim C}{\dim V} = \frac{k}{n}.$$

Later on, we'll look more closely at the negative effect of too much redundancy on the performance of certain codes.

DEFINITION 1.2. The **binary symmetric channel (BSC)** with **crossover probability p** takes a single binary input x (from the binary field \mathbb{F}) and switches it to $x + 1$ with probability p .

Throughout these notes, we shall assume that the BSC has crossover probability $p < 1/2$.

The BSC is typically viewed according to the following picture:



A fruitful way to view the BSC is that it takes an input vector from the code (a “codeword”), say $c \in C$, and adds a random noise vector $e \in V$ to c , producing the output vector $\hat{c} = c + e$. This random vector has a distribution dictated by the channel: if $e_0 = (\epsilon_1, \epsilon_2, \dots, \epsilon_n) \in V$, and if e_0 has l components not equal to zero, then

$$\text{Prob}(e = e_0) = p^l(1 - p)^{n-l}.$$

Now assume that $V = \mathbb{F}_n = \{(a_1, a_2, \dots, a_n) \mid a_i \in \mathbb{F}\}$, and let $C \subseteq V$ be a code. An important parameter associated with the code C is its **weight** (or **minimal weight**), $\text{wt}(C)$. First of all, if $v = (a_1, a_2, \dots, a_n) \in V$, set $\text{wt}(v) = |\{i \mid a_i \neq 0\}|$. Now set

$$\text{wt}(C) = \min_{0 \neq c \in C} \text{wt}(c).$$

As a result, if our code C has minimal weight $\text{wt}(C) = m$, then we will be able to *detect* any erroneously received word that differs in fewer than m coordinates from some codeword in C . Put differently, if the codeword c is sent and received as $\hat{c} = c + e$, then when $0 < \text{wt}(e) \leq m - 1$ we will know that there is an error in the received codeword \hat{c} , at which point we might ask for a retransmission.

From the above discussion, we see that for a code C of minimal weight m ,

$$\text{Prob}(\text{undetected error}) = \sum_{l=m}^n \binom{n}{l} p^l (1 - p)^{n-l} = \binom{n}{m} p^m + \text{higher degree terms}.$$

However, codes are useful not just for error *detection*; they can also be useful for error *correction*. However, for this to work, we need a mechanism for taking a received vector $v \in V$ and *deciding* which codevector was actually sent. In other words, a *decision rule* is really just a mapping

$$d : V \rightarrow C.$$

At this point, we introduce two commonly applied criteria for decision rules. First, however, it's reasonable to assume that we're trying to minimize the error in making our “decision” each time a vector is received, i.e., when we receive the vector v we want to minimize

$$\text{Prob}(\text{decision error} \mid v \text{ is received}).$$

Therefore, our unconditional decision error probability is

$$\text{Prob}(\text{decision error}) = \sum_{v \in V} \text{Prob}(\text{decision error} \mid v \text{ is received}) \text{Prob}(v \text{ is received}).$$

If $d : V \rightarrow C$ is our decision rule, then

$$\text{Prob}(\text{decision error} \mid v \text{ is received}) = 1 - \text{Prob}(d(v) \text{ is sent} \mid v \text{ is received}),$$

which gives us the error probability

$$\text{Prob}(\text{decision error}) = 1 - \sum_{v \in V} \text{Prob}(d(v) \text{ is sent} \mid v \text{ is received}) \text{Prob}(v \text{ is received}).$$

This error probability is a minimum precisely when each conditional probability $\text{Prob}(d(v) \text{ is sent} \mid v \text{ is received})$ is a *maximum*. Note, however, that this conditional probability is dependent on the *input distribution*, i.e., on the individual probabilities $\text{Prob}(c \text{ is sent})$, $c \in C$:

$$\begin{aligned} \text{Prob}(d(v) \text{ is sent} \mid v \text{ is received}) &= \frac{\text{Prob}(v \text{ is received} \mid d(v) \text{ is sent}) \text{Prob}(d(v) \text{ is sent})}{\text{Prob}(v \text{ is received})} \\ &= \frac{\text{Prob}(v \text{ is received} \mid d(v) \text{ is sent}) \text{Prob}(d(v) \text{ is sent})}{\sum_{c \in C} \text{Prob}(v \text{ is received} \mid c \text{ is sent}) \text{Prob}(c \text{ is sent})} \end{aligned}$$

where we have used *Bayes' Theorem* for inverting the conditional probability.

DEFINITION 1.3 (The Maximum A Posteriori Rule). *The decision rule $d : V \rightarrow C$ is called a **maximum a posteriori rule**, or **MAP rule** for short, if it maximizes each conditional probability*

$$\text{Prob}(d(v) \text{ is sent} \mid v \text{ is received}), \quad v \in V.$$

Part of the difficulty in constructing MAP decision rules is that they are based on “reverse probabilities,” whose calculations involve Bayes’ Theorem. As such they are dependent on the input distributions. We consider a couple of simple, but telling examples.

EXAMPLE 1. Consider the simplest possible example of a code, viz., take $C = \mathbb{F}_1 = \{0, 1\} = V$. Assume that the BSC crossover probability is p and that the input distribution is given by $\text{Prob}(0) = \epsilon$, $\text{Prob}(1) = 1 - \epsilon$. Therefore, we have that

$$\begin{aligned} \text{Prob}(0 \text{ is sent} \mid 0 \text{ is received}) &= \frac{\text{Prob}(0 \text{ is received} \mid 0 \text{ is sent}) \text{Prob}(0 \text{ is sent})}{\text{Prob}(0 \text{ is received})} \\ &= \frac{\text{Prob}(0 \text{ is received} \mid 0 \text{ is sent}) \text{Prob}(0 \text{ is sent})}{\sum_{i=0}^1 \text{Prob}(0 \text{ is received} \mid i \text{ is sent}) \text{Prob}(i \text{ is sent})} \\ &= \frac{(1-p)\epsilon}{(1-p)\epsilon + p(1-\epsilon)}. \end{aligned}$$

Similarly, one computes

$$\begin{aligned}\text{Prob}(1 \text{ is sent} \mid 0 \text{ is received}) &= \frac{p(1 - \epsilon)}{(1 - p)\epsilon + p(1 - \epsilon)} \\ \text{Prob}(0 \text{ is sent} \mid 1 \text{ is received}) &= \frac{p\epsilon}{p\epsilon + (1 - p)(1 - \epsilon)} \\ \text{Prob}(1 \text{ is sent} \mid 1 \text{ is received}) &= \frac{(1 - p)(1 - \epsilon)}{p\epsilon + (1 - p)(1 - \epsilon)}\end{aligned}$$

Therefore, if 0 is received, what does an MAP decision rule tell us to “decide” as to what was sent? From the above it is evident that if $(1 - p)\epsilon \leq p(1 - \epsilon)$, which is equivalent to saying that $\epsilon \leq p$, then we should decide that 1 was sent. Note that this same condition implies (since $p < 1/2$) that $p\epsilon \leq (1 - p)(1 - \epsilon)$ which means that if 1 is received, then we should also decide that 1 was sent. That is to say, if $\epsilon \leq p$, then *we should always decide that 1 was sent!* We leave the other cases to the reader to work out.

EXAMPLE 2. This time consider the one-dimensional code $C = \{(0, 0, 0), (1, 1, 1)\} \subseteq V = \mathbb{F}_3 = \{(a_1, a_2, a_3) \mid a_i \in \mathbb{F}\}$. As above, assume a crossover probability of p , and assume an input distribution

$$\text{Prob}(0, 0, 0) = \epsilon, \quad \text{Prob}(1, 1, 1) = 1 - \epsilon.$$

Assume that the vector $(1, 0, 0)$ was received. What is the best decision for what was sent ($(0, 0, 0)$ or $(1, 1, 1)$) according to an MAP decision rule? Again, we must use Bayes’ Theorem to calculate the necessary conditional probabilities:

$$\begin{aligned}&\text{Prob}((0, 0, 0) \text{ was sent} \mid (1, 0, 0) \text{ was received}) \\ &= \frac{\text{Prob}((1, 0, 0) \text{ was received} \mid (0, 0, 0) \text{ was sent})\text{Prob}((0, 0, 0) \text{ was sent})}{\text{Prob}((1, 0, 0) \text{ was received})} \\ &= \frac{p(1 - p)^2\epsilon}{p(1 - p)^2\epsilon + p^2(1 - p)(1 - \epsilon)},\end{aligned}$$

whereas,

$$\begin{aligned}&\text{Prob}((1, 1, 1) \text{ was sent} \mid (1, 0, 0) \text{ was received}) \\ &= \frac{\text{Prob}((1, 0, 0) \text{ was received} \mid (1, 1, 1) \text{ was sent})\text{Prob}((1, 1, 1) \text{ was sent})}{\text{Prob}((1, 0, 0) \text{ was received})} \\ &= \frac{p^2(1 - p)(1 - \epsilon)}{p(1 - p)^2\epsilon + p^2(1 - p)(1 - \epsilon)}.\end{aligned}$$

Thus, an MAP decision rule would tell us to decide that $(0, 0, 0)$ was sent if $p(1 - p)^2\epsilon \geq p^2(1 - p)(1 - \epsilon)$; since both $p, 1 - p, \epsilon > 0$, we see that this condition is equivalent to

$$\frac{1 - p}{p} \geq \frac{1 - \epsilon}{\epsilon}.$$

Again, the reader can complete this analysis.

The next decision rule is much easier to implement as it involves only “forward probabilities” and therefore is independent of the input distribution. This is the so-called *maximum likelihood decision rule*, as follows:

DEFINITION 1.4 (The Maximum Likelihood Rule). *The decision rule $d : V \rightarrow C$ is called a **maximum likelihood rule**, or **ML rule** for short, if it maximizes each conditional probability*

$$\text{Prob}(v \text{ is received} \mid d(v) \text{ is sent}), \quad v \in V.$$

Note that for the each of the above two examples, an ML decision rule is unique and is given by

$$d(0) = 0, \quad d(1) = 1, \quad \text{in Example 1;}$$

and

$$d(0, 0, 0) = d(1, 0, 0) = d(0, 1, 0) = d(0, 0, 1) = (0, 0, 0)$$

and

$$d(1, 1, 0) = d(1, 0, 1) = d(0, 1, 1) = d(1, 1, 1) = (1, 1, 1) \quad \text{in Example 2.}$$

Since only the forward probabilities of the channel are involved, we see that an ML decision rule doesn’t depend on the input distribution. However, as the above examples indicate, the two decision rules can differ. However, if the input distribution is uniform, then an ML decision rule is an MAP rule (and conversely):

LEMMA 1.1. *Assume that the input distribution is uniform, i.e., that for each $c \in C$,*

$$\text{Prob}(c \text{ is sent}) = \frac{1}{|C|}.$$

Then any ML decision rule $d : V \rightarrow C$ is also an MAP rule, and conversely.

PROOF. Indeed, for any vector $v \in V$, we have

$$\begin{aligned} \text{Prob}(d(v) \text{ is sent} \mid v \text{ is received}) &= \frac{\text{Prob}(v \text{ is received} \mid d(v) \text{ is sent})\text{Prob}(d(v) \text{ is sent})}{\text{Prob}(v \text{ is received})} \\ &= \frac{\text{Prob}(v \text{ is received} \mid d(v) \text{ is sent})}{|C|\text{Prob}(v \text{ is received})}. \end{aligned}$$

Therefore, we see that for fixed vector $v \in V$, $\text{Prob}(d(v) \text{ is sent} \mid v \text{ is received})$ is a maximum if and only if $\text{Prob}(v \text{ is received} \mid d(v) \text{ is sent})$ is a maximum.

The next result shows that decision rules based on minimal distance are maximum-likelihood decision rules (and conversely).

LEMMA 1.2. Let $C \subseteq V$ be a code. A decision rule $d : V \rightarrow C$ is a maximum-likelihood decision rule if and only if for each $v \in V$, $\text{wt}(v + d(v))$ is chosen to be a minimum.

PROOF. This is obvious, as for any pair $c \in C$, $v \in V$, we have

$$\text{Prob}(v \text{ is received} \mid c \text{ is sent}) = p^{\text{wt}(v+c)}(1-p)^{n-\text{wt}(v+c)},$$

which is a minimum if and only if $\text{wt}(v + c)$ is a minimum.

DEFINITION 1.5. Let $C \subseteq V$ be a code, and assume that $d : V \rightarrow C$ is a decision rule. If d satisfies the partial homomorphism property:

$$d(0) = 0, \quad d(v + c) = d(v) + c, \quad c \in C, v \in V,$$

then d is called a **standard array decision rule**.

Note that a standard array decision rule is uniquely determined by the values $d(v_1), d(v_2), \dots, d(v_r)$, where v_1, v_2, \dots, v_r is a set of coset representatives for C in V . Furthermore, a standard array decision rule $d : V \rightarrow V$ satisfies $d(c) = c$, for all $c \in C$.

LEMMA 1.3. Let $C \subseteq V$ be a code and let $v_1 = 0, v_2, \dots, v_r$ be a set of coset representatives of minimal weight, i.e., for each $i = 1, 2, \dots, r$, we have

$$\text{wt}(v_i) = \min_{c \in C} \text{wt}(v_i + c).$$

Then the standard array decision rule given by

$$d(v_i + c) = c, \quad i = 1, 2, \dots, r, \quad c \in C,$$

is an ML decision rule.

PROOF. This is virtually obvious since if $c' \in C$ is closer (in the sense of weight) to $v_i + c$ than is c , then $\text{wt}(v_i + c + c') < \text{wt}(v_i)$, contrary to v_i having minimal weight among elements in $v_i + C$.

We shall call a decision rule constructed as in Lemma 1.3 a *maximum likelihood standard array decision rule*; Lemma 1.3 guarantees that such a decision rule really is an ML rule. Note, however, that a given coset $v + C$ might not have a unique element of minimal weight, in which case an ML standard array decision rule is also not unique.

As a simple example, consider the 4-dimensional vector space $V = \mathbb{F}_4$ and take the code C to be the 2-dimensional subspace generated by $(1, 0, 1, 1)$, $(0, 1, 0, 1)$. The four cosets of C in V are listed below:

$$\begin{array}{cccc}
(0, 0, 0, 0) & (1, 0, 1, 1) & (0, 1, 0, 1) & (1, 1, 1, 0) \\
(1, 0, 0, 0) & (0, 0, 1, 1) & (1, 1, 0, 1) & (0, 1, 1, 0) \\
(0, 1, 0, 0) & (1, 1, 1, 1) & (0, 0, 0, 1) & (1, 0, 1, 0) \\
(0, 0, 1, 0) & (1, 0, 0, 1) & (0, 1, 1, 1) & (1, 1, 0, 0)
\end{array}$$

Note that each coset except the third contains a unique coset representative of minimal length. Thus, there are two maximum likelihood standard array decision rules $d_1, d_2 : V \rightarrow C$, and are determined by

$$d_1(0, 0, 0, 0), d_1(1, 0, 0, 0), d_1(0, 1, 0, 0), d_1(0, 0, 1, 0) = 0,$$

and

$$d_2(0, 0, 0, 0), d_2(1, 0, 0, 0), d_2(0, 0, 0, 1), d_1(0, 0, 1, 0) = 0.$$

We shall have occasion to refer to this example several more times in the sequel.

2 Bit Error Rate (BER)

We shall be sending k -bit (binary) messages across our noisy BSC with crossover probability p . We regard the k bit messages as vectors in the vector space $M = \mathbb{F}_k = \{(a_1, a_2, \dots, a_k) \mid a_i \in \mathbb{F}\}$. These messages are to be encoded as n -bit codewords via some “encoding map”

$$M = \mathbb{F}_k \xrightarrow{E} \mathbb{F}_n = V.$$

The image $C = E(M)$ is the code. Thus, corresponding to the k -bit message word m is the n -bit codeword $E(m)$. Owing to the noise in the channel, the received vector will have the form

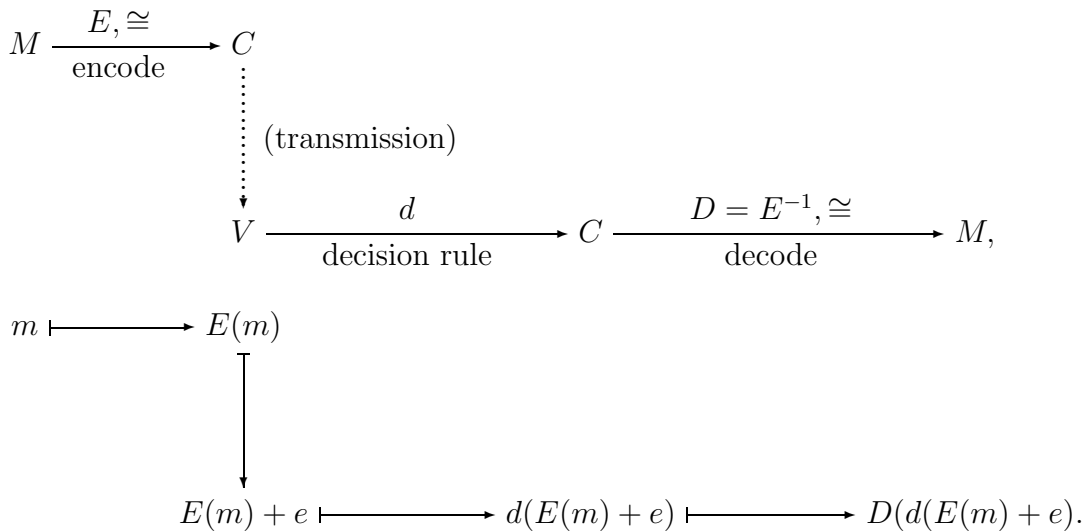
$$\widehat{E(m)} = E(m) + e,$$

where e is the random error vector having distribution

$$\text{Prob}(e = e_0) = p^l(1 - p)^{n-l},$$

where $l = \text{wt}(e_0)$.

The entire encoding/transmission/decision/decoding process can be viewed thus:



That is to say, the intended message m gets encoded, sent, received, decided upon, and decoded as the message $D(d(E(m) + e)) \in M$, where, as usual, e is the random error vector generated by the BSC.

Next, for any vector $m \in M$, denote by $m_{(i)}$ the i -th coordinate of m , i.e., $m = (m_{(1)}, \dots, m_{(k)}) \in M$, and consider the conditional probability

$$\text{Prob}(D(d(E(m) + e))_{(i)} \neq m_{(i)} \mid E(m) \text{ was sent}), \quad i = 1, 2, \dots, k.$$

This is simply the probability that the final message word disagrees with the intended message word in the i -th message bit. Put somewhat differently, we may define the random variables

$$X_i(m) = D(d(E(m) + e))_{(i)} + m_{(i)},$$

and consider

$$\text{Prob}(X_i(m) = 1),$$

for $i = 1, 2, \dots, k$.

We might ask the following questions about the random variables $X_i(m)$, $i = 1, 2, \dots, k$.

- (i) For fixed $m \in M$, are the random variables $X_i(m)$, $i = 1, 2, \dots, k$, identically distributed?
- (ii) For fixed $m \in M$, are the random variables $X_i(m)$, $i = 1, 2, \dots, k$, independent?
- (iii) For fixed i , $1 \leq i \leq k$, how do the $X_i(m)$ depend on the message vector $m \in M$?

We would certainly expect the answers to the above questions to depend on the decision rule $d: C \rightarrow V$.

If we average the probabilities $\text{Prob}(X_i(m) = 1)$ over $i = 1, 2, \dots, k$, then this defines the **conditional bit error rate**:

$$\begin{aligned} \text{BER}(m) &= \frac{1}{k} \sum_{i=1}^k \text{Prob}(X_i(m) = 1) \\ &= \frac{1}{k} \sum_{i=1}^k \text{Prob}(D(d(E(m) + e))_{(i)} \neq m_{(i)} \mid E(m) \text{ was sent}). \end{aligned}$$

In other words, given that the message m was the intended message, $k \cdot \text{BER}(m)$ is the expected number of bit errors in the message that actually turns up at the receiving end (after deciding and decoding). The (unconditional) **bit error rate**, is the weighted average over all possible messages¹:

¹This is equivalent to the *symbol error rate* P_{symb} given on page 20 of F.J. MacWilliams and N.J.A. Sloane's book, *The Theory of Error-Correcting Codes*, North-Holland Publishing Company, Amsterdam, 1978. While they don't explicitly say so, their definition is valid only for uniform input distributions.

$$\begin{aligned}
\text{BER} &= \frac{1}{k} \sum_{m \in M} \sum_{i=1}^k \text{Prob}(X_i(m) = 1) \text{Prob}(E(m) \text{ was sent}) \\
&= \frac{1}{k} \sum_{m \in M} \sum_{i=1}^k \text{Prob}(D(d(E(m) + e))_{(i)} \neq m_{(i)} \mid E(m) \text{ was sent}) \text{Prob}(E(m) \text{ was sent}).
\end{aligned}$$

Thus, $k\text{BER}$ is the expected number of message bit errors per transmission.

The above notion of bit error rate is what one might more properly call the **post-decoding bit error rate**, which is an average of the error probabilities in the message bits. If instead we consider the average of the error probabilities in the code bits, we would obtain what would be called the **post-decision bit error rate**:

$$\text{BER}_{pd} = \frac{1}{n} \sum_{m \in M} \sum_{i=1}^n \text{Prob}((d(E(m) + e))_{(i)} \neq E(m)_{(i)} \mid E(m) \text{ was sent}) \text{Prob}(E(m) \text{ was sent}).$$

In analogy with the above, $n\text{BER}_{pd}$ is the expected number of codebit errors per transmission.

Next, we show that when we use a standard array decision rule $d : C \rightarrow V$, the computations of BER and BER_{pd} can be simplified considerably.

PROPOSITION 2.1. *Let $C \subseteq V$, be a code with $\dim C = k$ and $\dim V = n$, and let $d : C \rightarrow V$ be a standard array decision rule. The the post-decision and post-decoding bit error rates are given by*

$$\text{BER} = \frac{1}{k} \sum_{e \in V} \text{wt}(Dd(e)) \text{Prob}(e), \text{ and}$$

$$\text{BER}_{pd} = \frac{1}{n} \sum_{e \in V} \text{wt}(d(e)) \text{Prob}(e).$$

PROOF. This is pretty easy. First of all, note that

$$\begin{aligned}
&\text{Prob}(D(d(E(m) + e))_{(i)} \neq m_{(i)} \mid E(m) \text{ was sent}) \\
&= \text{Prob}(D(E(m) + d(e))_{(i)} \neq m_{(i)} \mid E(m) \text{ was sent}) \\
&= \text{Prob}(m + D(d(e))_{(i)} \neq m_{(i)} \mid E(m) \text{ was sent}) \\
&= \text{Prob}(D(d(e))_{(i)} \neq 0 \mid E(m) \text{ was sent}).
\end{aligned}$$

However, the error vector $e \in V$ is generated by the BSC independently of which encoded message $E(m)$ was sent, thus

$$\text{Prob}(D(d(e))_{(i)} \neq 0 \mid E(m) \text{ was sent}) = \text{Prob}(D(d(e))_{(i)} \neq 0).$$

Therefore,

$$\begin{aligned}
\text{BER} &= \frac{1}{k} \sum_{m \in M} \sum_{i=1}^k \text{Prob}(D(d(E(m) + e))_{(i)} \neq m_{(i)} \mid E(m) \text{ was sent}) \text{Prob}(E(m) \text{ was sent}) \\
&= \frac{1}{k} \sum_{m \in M} \sum_{i=1}^k \text{Prob}(D(d(e))_{(i)} \neq 0) \text{Prob}(E(m) \text{ was sent}) \\
&= \frac{1}{k} \sum_{i=1}^k \text{Prob}(D(d(e))_{(i)} \neq 0) \\
&= \frac{1}{k} \sum_{e \in V} \text{wt}(Dd(e)) \text{Prob}(e)
\end{aligned}$$

The proof of the corresponding recipe for BER_d is entirely similar. In other words, we see that $k \cdot \text{BER}$ is the expected weight (measured in M) of the random vector $Dd(e)$, $e \in V$. Similarly, $k \cdot \text{BER}_d$ is the expected weight (measured in V) of the random vector $d(e)$, $e \in V$.

It would be of interest to determine under what conditions $\text{BER} = \text{BER}_{pd}$. In general, one wouldn't expect them to agree, if only because BER at least ostensibly depends upon the encoding $E : M \rightarrow C$, as well as on the decision rule $d : V \rightarrow C$, whereas BER_{pd} depends only upon the decision rule. However, if one uses *systematic encoding* (to be explained below), then one might reasonably inquire as to whether one might have $\text{BER} = \text{BER}_{pd}$, say under the assumption of an ML standard array decision rule.

In fact, this is what initially spurred my interest in this endeavor, for in my quest for the BER of the Hamming codes, I was referred by Michele Eile to the "standard reference" by J. H. van Lint, Coding Theory, *Lecture Notes in Mathematics*, vol. 201, Springer-Verlag, New York, 1973, pp. 25–26. However, van Lint computes $n\text{BER}_{pd}$ for the Hamming codes and not $k\text{BER}$. I am still searching for a computation of BER , (or $k\text{BER}$) although it appears that for the Hamming codes (under the assumption of systematic encoding), $\text{BER} = \text{BER}_{pd}$. I'll give evidence for this in the next section.

EXAMPLE. We refer again to the example $C \subseteq V$ given on page 6 and compute its post-decision BER relative to the two ML standard array decision rules $d_1, d_2 : V \rightarrow C$. We have

$$\begin{aligned}
4\text{BER}_{pd} &= \sum_{e \in V} \text{wt}(d_1(e)) \text{Prob}(e) \\
&= \sum_{j=0}^4 \sum_{e \in V, \text{wt}(e)=j} \text{wt}(d_1(e)) \text{Prob}(e) \\
&= 2p(1-p)^3 + 17p^2(1-p)^2 + 10p^3(1-p) + 3p^4.
\end{aligned}$$

The same result holds for the decision rule $d_2 : V \rightarrow C$.

DEFINITION 2.1. Let $C \subseteq V = \mathbb{F}_n$ be a code. An encoding scheme $E : M = \mathbb{F}_k \xrightarrow{\cong} C$ is called **systematic**, or is a **row echelon form encoding scheme**, if and only if the $k \times n$ matrix G having rows r_1, r_2, \dots, r_k is in row echelon form, where

$$E(0, 0, \dots, 1, 0, \dots, 0) = r_i \in C.$$

1 in position i

When this happens, a permutation of the columns can be applied to bring the matrix into the form

$$G = \left[\begin{array}{c|c} I_{k \times k} & A_{k \times n-k} \end{array} \right]$$

and that a codeword $(a_1, a_2, \dots, a_n) \in C$ will be decoded as

$$D(a_1, a_2, \dots, a_n) = (a_1, a_2, \dots, a_k) \in M.$$

To see how different encoding schemes can lead to different BERs, even with respect to the same decision rule, consider once again the example of the two-dimensional code, given on page 6. Thus, $C \subseteq V = \mathbb{F}_4$ has basis $\{r_1 = (1, 0, 1, 1), r_2 = (0, 1, 0, 1)\}$ and the encoding scheme $E : (a_1, a_2) \mapsto a_1 r_1 + a_2 r_2$ is systematic. With respect this choice of encoding and with respect to the ML standard array decision rule $d = d_1$ given on page 7, one has

$$\begin{aligned} 2\text{BER} &= \sum_{j=0}^4 \sum_{e \in V, \text{wt}(e)=j} \text{wt}(Dd(e)) \text{Prob}(e) \\ &= p(1-p)^3 + 9p^2(1-p)^2 + 5p^3(1-p) + p^4 \approx p + 6p^2, \end{aligned}$$

for small enough p .

On the other hand, were one to take the non-systematic encoding $E' : (a_1, a_2) \mapsto a_1 r'_1 + a_2 + r'_2$, where

$$r'_1 = r_1 = (1, 0, 1, 1), \quad r'_2 = r_1 + r_2 = (1, 1, 1, 0),$$

then the corresponding BER (relative to $d = d_1$) is given by

$$\begin{aligned} 2\text{BER} &= \sum_{j=0}^4 \sum_{e \in V, \text{wt}(e)=j} \text{wt}(D'd(e)) \text{Prob}(e) \\ &= 2p(1-p)^3 + 7p^2(1-p)^2 + 6p^3(1-p) + p^4 \approx 2p + p^2, \end{aligned}$$

for small p . Therefore, we conclude that, at least for small enough values of p (i.e., for a good enough BSC), we see that the BER computed in terms of the ML standard array decision rule $d = d_1$ and systematic encoding is less than the corresponding BER computed in terms of the non-systematic encoding scheme $E' : M \rightarrow C$.²

²The equivocation “for small enough p ,” turns out not to be necessary, since one can show in this example that the BER for systematic encoding E is less than the BER with respect to $E' : M \rightarrow C$ for all p between 0 and 1/2.

REMARK. Note that neither of the post-decoding bit error rates agree with the post-decision BER given on page 10.

The following theorem would be highly desirable—I’ll state it as a conjecture. It should be known, but I’ve not seen any relevant discussions.

CONJECTURE. Let $C \subseteq V = \mathbb{F}_n$ be a code and fix an ML standard array decision rule $d : V \rightarrow C$. Let $E, E' : M = \mathbb{F}_k \xrightarrow{\cong} C$ be encoding schemes with E systematic. If $\text{BER}_E, \text{BER}_{E'}$ are the corresponding bit error rates, then $\text{BER}_E \leq \text{BER}_{E'}$ for small enough p .^{3 4}

3 The Hamming Codes

In principle, the Hamming Codes are very easy to describe. To this end, we fix an l -dimensional vector space W over the binary field \mathbb{F}_2 . Let \mathcal{P} be the set of nonzero vectors in W and let $V = \mathbb{F}\langle\mathcal{P}\rangle$ be the vector space with basis \mathcal{P} . Thus we have the “tautological map” $\tau : V \rightarrow W$ determined by $v \mapsto v$, $v \in \mathcal{P}$. We set $\mathbf{H}(W)$ to be the kernel of $V \rightarrow W$, and call it the *Hamming code on W* .⁵ Thus the Hamming code on W fits into an exact sequence

$$0 \longrightarrow \mathbf{H}(W) \longrightarrow \mathbb{F}\langle\mathcal{P}\rangle \xrightarrow{\tau} W \longrightarrow 0,$$

from which it follows that $\dim \mathbf{H}(W) = 2^l - l - 1$. Thus, it is obvious that the minimal weight of $\mathbf{H}(W)$ is 3, since no two (or fewer) nonzero vectors in W can be linearly dependent. (Note that the vector space V can be identified with the Boolean group (with symmetric difference as the operation) on the set of nonzero vectors of W .)

DEFINITION 3.1. Let $C \subset V$ be an (n, k) linear block code of minimal weight δ . We say that C is a **perfect** code if there exists $r < \delta$ such that

$$V = \bigcup_{c \in C} B_c(r), \quad \text{disjoint union,}$$

where $B_c(r) = \{v \in V \mid \text{wt}(v + c) \leq r\}$.

For such a code we see that the ML decision rule is uniquely determined: if $v \in V$ we take $d(v) = c$, where $c \in C$ is the unique vector living in $B_v(r)$, where r is chosen in accordance with the above. Furthermore, as a result of *Lemma 1.3*, we conclude that the ML decision rule is necessarily a standard array decision rule.

This is all relevant, because of

³Again, this restriction on p might not be necessary.

⁴In the definition of BER in MacWilliams-Sloane, it is tacitly assumed that the encoding scheme is systematic.

⁵If $n = 2^l - 1$ and $k = n - l$, we sometimes call $\mathbf{H}(W)$ the (n, k) -Hamming code.

LEMMA 3.1. *The Hamming code $C = \mathbf{H}(W) \subseteq V$ is a perfect code of minimal weight 3.*

PROOF. Since C has minimal weight 3, it already follows that for all $c \neq c'$ in C , we have $B_c(1) \cap B_{c'}(1) = \emptyset$. Next, let $\dim W = l$; it is clear that for all $c \in C$, $|B_c(1)| = n + 1 = 2^l$. Therefore,

$$\left| \bigcup_{c \in C} B_c(1) \right| = (n + 1)|C| = (2^l)(2^{n-l}) = 2^n = |V|.$$

4 Second-Order BER for the Hamming Codes

Let W be an l -dimensional vector space over the binary field \mathbb{F} , and let $C = \mathbf{H}(W) \subseteq V = \mathbb{F}\langle \mathcal{P} \rangle$, be the corresponding Hamming code, where, as above, \mathcal{P} is the set of nonzero vectors of W . Relative to the unique ML standard array decision rule $d : V \rightarrow C$ and a systematic encoding scheme $E : M = \mathbb{F}_k \rightarrow C$, $k = 2^l - l - 1$, we shall compute the second order term (= coefficient of p^2) in the bit error rate. Indeed, this makes sense, as *Proposition 2.1* shows that the bit error rate, as well as the post-decision bit error rate are polynomials in the crossover probability p of the BSC.

We wish to give an intrinsic characterization of systematic encoding schemes $E : M \rightarrow C$. If $n = 2^l - 1$, then any fixed ordering of the vectors in \mathcal{P} gives an isomorphism $\mathbb{F}_n \rightarrow V = \mathbb{F}\langle \mathcal{P} \rangle$. For convenience, if $S \subseteq \mathcal{P}$ is a subset, let $[S] = \sum_{s \in S} s \in W$. Next, let

$\mathcal{B} = \{w_1, w_2, \dots, w_l\}$ be a basis of W . For each $j \geq 2$, let $S_{ij} \subseteq \mathcal{B}$, $i = 1, 2, \dots, \binom{l}{j}$, be the distinct subsets of size j in \mathcal{B} . Note that

$$\sum_{j=2}^l \binom{l}{j} = 2^l - l - 1,$$

and that the vectors $[S_{ij}] \in W$ are all distinct and none is in \mathcal{B} . Next, for each nonzero vector $w \in \mathcal{P}$, let $\mu_w : \mathbb{F} \rightarrow V = \mathbb{F}\langle \mathcal{P} \rangle$ be the w -th coordinate function. If $\mathcal{Q} \subseteq \mathcal{P}$ define the element $[[\mathcal{Q}]] \in V$ by setting

$$[[\mathcal{Q}]] = \sum_{q \in \mathcal{Q}} \mu_q(1) \in V.$$

Finally, define the vectors $r_{ij} \in V$, $2 \leq j \leq l$, $1 \leq i \leq \binom{l}{j}$ by setting

$$r_{ij} = [[\{[S_{ij}]\} \cup S_{ij}]].$$

For example, consider the (7, 4)-Hamming code $\mathbf{H}(W)$, and so W is 3-dimensional and has basis $\mathcal{B} = \{w_1, w_2, w_3\}$. Take the ordering of the j -element subsets of \mathcal{B}

to be $\{w_1, w_2\}, \{w_1, w_3\}, \{w_2, w_3\}, \{w_1, w_2, w_3\}$. Then relative to this ordering, the elements $r_{ij} \in V$ are the row vectors

$$\begin{aligned} r_{12} &= (1, 0, 0, 0, 1, 1, 0), \\ r_{22} &= (0, 1, 0, 0, 1, 0, 1), \\ r_{32} &= (0, 0, 1, 0, 0, 1, 1), \\ r_{13} &= (0, 0, 0, 1, 1, 1, 1). \end{aligned}$$

It is clear that these vectors form a basis of $C = \mathbf{H}(W)$ and that the corresponding matrix is in row echelon form. Thus, the encoding scheme

$$E : (1, 0, 0, 0) \mapsto r_{12}, (0, 1, 0, 0) \mapsto r_{22}, (0, 0, 1, 0) \mapsto r_{32}, (0, 0, 0, 1) \mapsto r_{13}$$

is systematic. Finally, it is not hard to see that any systematic encoding scheme must arise in the above fashion.

We let $\text{BER}^{(2)}$, $\text{BER}_{pd}^{(2)}$ be the second-order bit error rate and post-decision bit error rate, respectively, of the Hamming code using the unique ML standard array decision rule $d : V \rightarrow C$ and a systematic encoding scheme $E : M \rightarrow C$. Thus,

$$k\text{BER}^{(2)} = \sum_{e \in V, \text{wt}(e)=2} \text{wt}(Dd(e))\text{Prob}(e) = \sum_{e \in V, \text{wt}(e)=2} \text{wt}(Dd(e))p^2(1-p)^{n-2},$$

and

$$n\text{BER}_{pd}^{(2)} = \sum_{e \in V, \text{wt}(e)=2} \text{wt}(d(e))\text{Prob}(e) = \sum_{e \in V, \text{wt}(e)=2} \text{wt}(d(e))p^2(1-p)^{n-2}.$$

We prove below that at least the second order bit error rates do agree:

THEOREM 4.1. *For the Hamming code $\mathbf{H}(W)$ with decision rule and encoding scheme as above, $\text{BER}^{(2)} = \text{BER}_{pd}^{(2)} = \frac{3}{2}(n-1)$.*

PROOF. Perhaps we should note first that neither BER nor BER_{pd} contain nonzero linear terms in p . This is because error vectors of weight 1 are closest to the zero vector in C and hence if $\text{wt}(e) = 1$, then $d(e) = 0$, i.e., such errors get “corrected.” Next, note that if $\text{wt}(e) = 2$, then $d(e) \in C$ is necessarily a vector of weight 3. Therefore,

$$\frac{1}{n} \sum_{e \in V, \text{wt}(e)=2} \text{wt}(d(e)) = \frac{3}{n} \binom{n}{2} = \frac{3}{2}(n-1).$$

On the other hand, note that each codeword $c \in C$ of weight 3 is $d(e)$ for precisely three error vectors $e \in V$ of weight 2. Therefore,

$$\frac{1}{k} \sum_{e \in V, \text{wt}(e)=2} \text{wt}(Dd(e)) = \frac{3}{k} \sum_{c \in C, \text{wt}(c)=3} \text{wt}(D(c)).$$

Therefore, we have reduced the problem to that of showing

$$\sum_{c \in C, \text{wt}(c)=3} \text{wt}(D(c)) = \frac{k}{2}(n-1).$$

We now recall the basis elements $\{r_{ij} \mid 2 \leq j \leq l, 1 \leq i \leq j\}$ of $C = \mathbf{H}(W)$. Clearly the elements of C such that $\text{wt}(D(c)) = 1$ are the basis elements r_{ij} . However, those of weight 3 (in C) are precisely the basis vectors $r_{12}, r_{22}, \dots, r_{\binom{l}{2}2}$, i.e., the number of vectors $c \in C$ of weight 3 and such that $\text{wt}(D(c)) = 1$ is $\binom{l}{2}$. Next, the vectors in C with $\text{wt}(D(c)) = 2$ are of the form $r_{ij} + r_{uv}$, where $\{i, j\} \neq \{u, v\}$. Such a vector has weight 3 in C precisely when $|S_{ij}| = |S_{uv}| + 1$, and $S_{ij} \supseteq S_{uv}$ or when $|S_{ij}| = |S_{uv}| - 1$, and $S_{ij} \subseteq S_{uv}$. The number of such subsets can be enumerated thus:

$$\binom{l}{3} \binom{3}{2} + \binom{l}{4} \binom{4}{3} + \dots + \binom{l}{l} \binom{l}{l-1} = \sum_{s=3}^l \binom{l}{s} \binom{s}{s-1}.$$

This quantity can be computed fairly easily. We have, by the *Binomial Theorem*, that

$$(1+x)^l = \sum_{s=0}^l \binom{l}{s} x^s,$$

and so

$$l(1+x)^{l-1} = \frac{d}{dx}(1+x)^l = \sum_{s=1}^l \binom{l}{s} s x^{s-1} = \sum_{s=1}^l \binom{l}{s} \binom{s}{s-1} x^{s-1}.$$

Therefore,

$$\sum_{s=1}^l \binom{l}{s} \binom{s}{s-1} = l(1+1)^{l-1} = l2^{l-1},$$

from which it follows that

$$\begin{aligned} \sum_{s=3}^l \binom{l}{s} \binom{s}{s-1} &= l2^{l-1} - \binom{l}{2} \binom{2}{1} - \binom{l}{1} \binom{1}{0} \\ &= l2^{l-1} - l(l-1) - l \\ &= l(2^{l-1} - l). \end{aligned}$$

In other words, the number of codewords $c \in C$ of weight 3 with $\text{wt}(D(c)) = 2$ is $l(2^{l-1} - l)$. Finally, the number of codewords $c \in C$ of weight 3 and having $\text{wt}(D(c)) = 3$ is equal to

$$(\# \text{ codewords in } C \text{ of weight 3}) - \binom{l}{2} - l(2^{l-1} - l).$$

Clearly, the number of codewords of weight 3 in C is equal to the number of 2-dimensional subspaces in W , which in turn is given by the Gaussian coefficient

$$\begin{bmatrix} l \\ 2 \end{bmatrix}_2 = \frac{(2^l - 1)(2^l - 2)}{(2^2 - 1)(2^2 - 2)} = \frac{1}{3} \binom{n}{2}.$$

Therefore, the number of codewords $c \in C$ of weight 3 and having $\text{wt}(D(c)) = 3$ is equal to

$$\frac{1}{3} \binom{n}{2} - \binom{l}{2} - l(2^{l-1} - l).$$

Putting all this together, we have

$$\begin{aligned} \sum_{c \in C, \text{wt}(c)=3} \text{wt}(D(c)) &= \binom{l}{2} + 2l(2^{l-1} - l) + 3 \left[\frac{1}{3} \binom{n}{2} - \binom{l}{2} - l(2^{l-1} - l) \right] \\ &= \frac{k}{2} (n - 1) \quad \text{after some calculation!} \end{aligned}$$

This completes the proof.

5 van Lint's Calculation of BER_{pd}

The calculation of BER_{pd} for the Hamming codes, first given by van Lint⁶ is actually fairly easy. To this end let $C = \mathbf{H}(W)$ be an (n, k) -Hamming code, where $n = 2^l - 1$, $k = 2^l - l - 1$. For each integer $i = 0, 1, 2, \dots, n$, let

$$A_i = \#(\text{codewords of weight } i).$$

Thus, we have already seen that $A_0 = 1$, $A_1, A_2 = 0$, $A_3 = \frac{1}{3} \binom{n}{2}$.

We have

$$\begin{aligned} n\text{BER}_{pd} &= \sum_{e \in V} \text{wt}(d(e)) \text{Prob}(e) \\ &= \sum_{c \in V} \sum_{e \in B_c(1)} \text{wt}(d(e)) \text{Prob}(e) \\ &= \sum_{i=0}^n \sum_{\substack{c \in C \\ \text{wt}(c) = i}} \sum_{e \in B_c(1)} \text{wt}(d(e)) \text{Prob}(e). \end{aligned}$$

Next, for a fixed codevector $c \in C$ of weight i , note that

$$\begin{aligned} \sum_{e \in B_c(1)} \text{wt}(d(e)) \text{Prob}(e) &= i^2 p^{i-1} (1-p)^{n-i+1} + ip^i (1-p)^{n-i} + i(n-i)p^{i+1} (1-p)^{n-i-1} \\ &= iP(p, i), \end{aligned}$$

where we have set

$$P(p, i) = ip^{i-1} (1-p)^{n-i+1} + p^i (1-p)^{n-i} + (n-i)p^{i+1} (1-p)^{n-i-1}.$$

Therefore, it follows that

$$\begin{aligned} n\text{BER}_{pd} &= \sum_{i=0}^n iA_i P(p, i) \\ &= \sum_{i=0}^n iA_i [ip^{i-1} (1-p)^{n-i+1} + p^i (1-p)^{n-i} + (n-i)p^{i+1} (1-p)^{n-i-1}]. \end{aligned}$$

⁶Lecture Notes in Mathematics," vol. 201, Springer-Verlag, New York, 1973, pp. 25–26.

Therefore, we see that for the Hamming codes, the post-decision bit error rate depends solely on the so-called *weight enumerator polynomial*, which is given by

$$A(x) = \sum_{i=0}^n A_i x^i.$$

Van Lint has also, computed $A(x)$; we shall sketch his development. We begin by setting $\mathcal{A}_i = \{c \in C \mid \text{wt}(c) = i\}$. Notice that the weight i vectors $v \in V$ come from three sources:

- (1) Those that are already codevectors in \mathcal{A}_i .
- (2) Those that are distance 1 from codevectors in \mathcal{A}_{i+1} ; note that each codevector gives rise to $\binom{i+1}{i} = i+1$ vectors of weight i .
- (3) Those that are distance 1 from codevectors in \mathcal{A}_{i-1} .

From the above, it follows that

$$\binom{n}{i} = |\mathcal{A}_i| + (i+1)|\mathcal{A}_{i+1}| + (n-i+1)|\mathcal{A}_{i-1}|,$$

and so

$$\binom{n}{i} = A_i + (i+1)A_{i+1} + (n-i+1)A_{i-1}.$$

Multiply both sides of this equation by x^i and sum over $i = 0, 1, \dots, n+1$:

$$\sum_{i=0}^{n+1} \binom{n}{i} x^i = \sum_{i=0}^{n+1} A_i x^i + \sum_{i=0}^{n+1} (i+1)A_{i+1} x^i + \sum_{i=0}^{n+1} (n-i+1)A_{i-1} x^i,$$

and so

$$\sum_{i=0}^n \binom{n}{i} x^i = \sum_{i=0}^n A_i x^i + \frac{d}{dx} \sum_{i=0}^{n+1} A_{i+1} x^{i+1} + n \sum_{i=0}^{n+1} A_{i-1} x^i - \sum_{i=0}^{n+1} (i-1)A_{i-1} x^i.$$

This implies, of course, that

$$\begin{aligned} (1+x)^n &= A(x) + A'(x) + nx A(x) - x^2 \sum_{i=0}^n i A_i x^{i-1} \\ &= A(x) + A'(x) + nx A(x) - x^2 A'(x), \end{aligned}$$

which can be written as the first-order Bernoulli equation

$$A'(x) + \left(\frac{1+nx}{1-x^2} \right) A(x) = \frac{(1-x)^n}{1-x^2}.$$

Recall that the first-order Bernoulli equation $y' + p(x)y = q(x)$ is solved by multiplying through by the integrating factor $u(x) = e^{\int p(x)dx}$, with the solution being

$$y = \frac{1}{u(x)} \int u(x)q(x)dx.$$

If we apply this to the above, the final result is that

$$A(x) = \frac{1}{n+1}(1+x)^n + \frac{n}{n+1}(1+x)^{(n-1)/2}(1-x)^{(n+1)/2}.$$

We return now to the computation of BER_{pd}. If we set $q = 1 - p$, then we have

$$\begin{aligned} n\text{BER}_{pd} &= q^n \sum_{i=0}^n iA_i \{x^i + (n-i)x^{i+1} + ix^{i-1}\} \\ &= q^n \{((n-1)x^2 + x + 1)A'(x) + (x - x^3)A''(x)\} \end{aligned}$$

After some calculation, this ultimately boils down to

$$\begin{aligned} n\text{BER}_{pd} &= \frac{n}{n+1} \left((n-1) \frac{p^2}{1-p} + 1 \right) [1 - (1 + (n-1)p)(1-2p)^{(n-1)/2}] \\ &+ \frac{p(1-2p)}{1-p} \cdot \frac{n(n-1)}{n+1} [1 + (np^2 - p^2 + 4p - 1)(1-2p)^{(n-3)/2}] \end{aligned}$$

In order to interpret the limiting value of the above, note first that the expression $\alpha = np$ represents the expected number of codebit errors in unencoded transmission of n codebits. So we fix $\alpha = np$ and let $n \rightarrow \infty$, $p \rightarrow 0$ in the above expression of $n\text{BER}_{pd}$ and find that

$$\begin{aligned} n\text{BER}_{pd} &\rightarrow [1 - (1 + \alpha)e^{-\alpha}] + \alpha(1 - e^{-\alpha}) \\ &= \alpha + [1 - (1 + 2\alpha)e^{-\alpha}]. \end{aligned}$$

In particular, if $(1 + 2\alpha)e^{-\alpha} > 1$ (which is roughly saying that $\alpha > 1.2564$), we see that asymptotically $n\text{BER}_{pd}$ is greater than α , i.e., the Hamming codes *do worse than with no coding at all!*

6 BER = BER_{pd} for the (7, 4)-Hamming Code

In the previous section, we saw that if $A(x) = \sum_{i=0}^n A_i x^i$ was the weight enumerator for the Hamming code C , then the post-decision bit error rate is given by

$$\text{BER}_{pd} = \frac{1}{n} \sum_{i=0}^n iA_i P(p, i)$$

where,

$$P(p, i) = [ip^{i-1}(1-p)^{n-i+1} + p^i(1-p)^{n-i} + (n-i)p^{i+1}(1-p)^{n-i-1}].$$

On the other hand, relative to decoding $D : C \rightarrow M$, we have that the post-decoding bit error rate is

$$\begin{aligned}
\text{BER} &= \frac{1}{k} \sum_{e \in V} \text{wt}(D(d(e))) \text{Prob}(e) \\
&= \frac{1}{k} \sum_{c \in C} \sum_{e \in B_c(1)} \text{wt}(D(d(e))) \text{Prob}(e) \\
&= \frac{1}{k} \sum_{i=0}^n \sum_{\substack{c \in C \\ \text{wt}(c) = i}} \sum_{e \in B_c(1)} \text{wt}(D(d(e))) \text{Prob}(e) \\
&= \sum_{i=0}^n \sum_{\substack{c \in C \\ \text{wt}(c) = i}} \text{wt}(D(c)) P(p, i) \\
&= \sum_{i=0}^n P(p, i) \sum_{\substack{c \in C \\ \text{wt}(c) = i}} \text{wt}(D(c)).
\end{aligned}$$

Therefore, we see that $\text{BER}_{pd} = \text{BER}$ provided that we can show

$$\sum_{\substack{c \in C \\ \text{wt}(c) = i}} \text{wt}(D(c)) = \frac{ki}{n} A_i,$$

for each $i = 0, 1, \dots, n$.

For the (7, 4)-Hamming code, the weight enumerator polynomial is $A(x) = 1 + 7x^3 + 7x^4 + x^7$. Verifying that

$$\sum_{\substack{c \in C \\ \text{wt}(c) = 3}} \text{wt}(D(c)) = 12,$$

$$\sum_{\substack{c \in C \\ \text{wt}(c) = 4}} \text{wt}(D(c)) = 16,$$

and that

$$\sum_{\substack{c \in C \\ \text{wt}(c) = 7}} \text{wt}(D(c)) = 4,$$

is entirely routine and can be left to the reader.

7 Performance of the Hamming Codes

This final section can best be thought of as an “engineering appendix,” as it documents how electrical engineers view the performance of codes. First of all we need a way to compute the crossover probability of error for a BSC. This depends on a number of factors, including the mode of binary signaling, the mode of reception, the energy E_b per sent bit and the white noise spectral density, N_0 . A common assumption is to use what is called “bipolar signaling,” and invoke a theorem of electrical engineering which asserts that the probability of error (=crossover probability) is *minimized* precisely when the “matched filter” reception design⁷ is used, with the resulting error probability given by

$$p = Q\left(\sqrt{\frac{2E_b}{N_0}}\right),$$

and where the Q function is defined by

$$Q(z) = \frac{1}{\sqrt{2\pi}} \int_z^{\infty} e^{-\lambda^2/2} d\lambda.$$

In applying this to, say, the (7, 4)-Hamming code, we must realize that the above calculation is predicated on having invested E_b joules in one bit. However, in the (7, 4)-Hamming code there is redundancy to the extent that seven actual electronic “bits” are sent for every four message bits. Since the BER is reflective of the message bits, then we must realize that in order to dedicate 1 Joule of energy to a message bit, we need to put in $\frac{7}{4}$ Joules into each codebit (the physically transmitted bit). Therefore, if we write the bit error rate as a polynomial in the crossover probability p :

$$\text{BER} = B(p),$$

then in terms of E_b/N_0 (the “engineering standard”) we would use

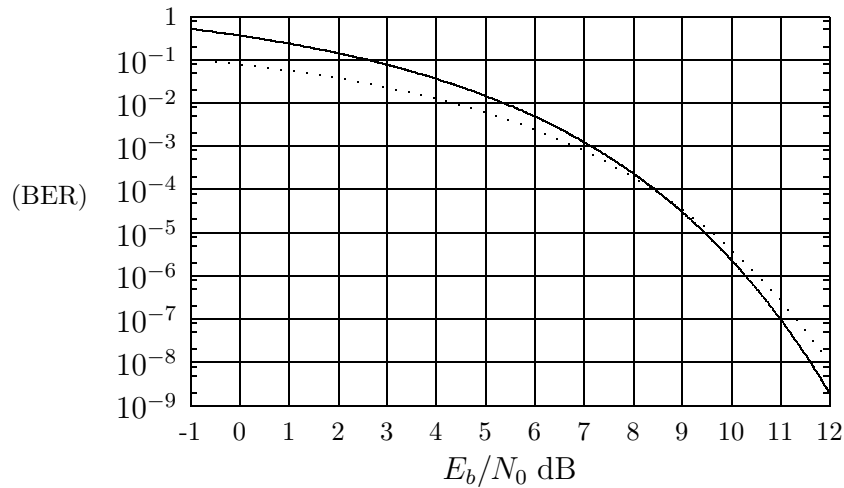
$$\text{BER} = B\left(Q\left(\sqrt{8E_b/7N_0}\right)\right)$$

for the computation.

In the graph below, the second-order approximation $\text{BER} \approx 9p^2 = 9\left[Q\left(\sqrt{8E_b/7N_0}\right)\right]^2$, taken from Theorem 4.1 has been used. The resulting performance graph, with the comparison taken against the unencoded transmission is as below:

⁷Such a filter is characterized by the fact that its impulse response is matched to a (reverse copy of) the known input signal.

Performance of (7, 4)-Hamming Code vs. Unencoded Transmission (···)



The way engineers read this graph is by saying, for example, that if one wants a bit error rate of no worse than 10^{-8} , then one must arrange to send signals that are at least 11.5 dB over the background white noise for the Hamming codes and at least 12 dB over the background white noise for the unencoded transmission. Put differently, at a BER of 10^{-8} we see is roughly a 1/2 dB *gain* over unencoded transmission.

Below, we have given the performance graph for the (15, 11)-Hamming code. In this case, at a BER of 10^{-8} we see approximately a 1.5 dB gain as compared with the unencoded transmission.

Performance of (15, 11)-Hamming Code vs. Unencoded Transmission (···)

