Stony Brook University
MAT 312/AMS 351 – Fall 2010
Notes on Hamming Codes

1. Error-detecting matrices.

Suppose a group code is generated by a matrix

$$G = \left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{array}\right).$$

We can write this matrix as

$$G = (I_4|A)$$

where A is the 4×3 matrix encoding the check bits.
Let us construct another matrix using A:

$$H = \left(\frac{A}{I_3}\right).$$

In this case,

$$H = \left(\begin{array}{ccc} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right).$$

In general if our code is $f : \mathbf{B}^m \to \mathbf{B}^n$, then G is of the form $G = (I_m|A)$ and H of the form $H = \left(\frac{A}{I_{n-m}}\right)$.

Remark: The product GH is a 4×3 (in general, $m \times (n-m)$) matrix of zeroes. What happens is that during the multiplication each element in A gets added to itself, giving 0. Schematically

$$GH = (I_m|A) \cdot \left(\frac{A}{I_{n-m}}\right) = I_m A + A I_{n-m} = A + A = \mathbf{0}$$

where here $\mathbf{0}$ represents the $m \times (n-m)$ matrix of zeroes.

This means that H, applied on the right to any one of the code-words, must give the length $n-m$ 0-vector: $(0,\ldots,0)$, since a code-word is of the form wG for some $w \in \mathbf{B}^m$, and $(wG)H = w(GH) = (0,\ldots,0)$.

So the matrix H can be used to test if an error has occurred in the transmission of a code-word $c \to c'$: if $c'H \neq (0,\ldots,0)$, then an error was made in the transmission.

2. Hamming codes: error-correcting matrices

In Hamming codes the matrix A is constructed in such a way that the vector $c'H$ not only signals that an error has occurred, but identifies the erroneous bit. So H can correct the error as well as detecting it.

The scheme of check-bits in a Hamming code is derived from the binary representation of the *position* of the bits being checked.

- Check-bit 1 is a parity check on all bits whose position number (in binary) is of the form xxx1: bits in position 3 = 11, 5 = 101, 7 = 111, etc.
- Check-bit 2 is a parity check on all bits whose position number is of the form xx1x (has a 1 in next-to-last position): bits 3 = 11, 6 = 110, 7 = 111, etc.
- Check-bit 4 is a parity check on all bits whose position number is of the form x1xx: bits 5 = 101, 6 = 110, 7 = 111, etc.
- This pattern can be extended to handle data words of arbitrary length. Here we will consider data words of length 4, so these first three check bits will suffice.

In order to implement the Hamming code by a matrix G as above, i.e. where the check bits come after the data bits, we reorder the positions in the code-word:

$$\textit{1}, \textit{2}, 3, \textit{4}, 5, 6, 7 \to 3, 5, 6, 7, \textit{4}, \textit{2}, \textit{1}.$$

With that ordering, the matrix G implementing our 3 check-bits becomes

$$G = \left(\begin{array}{c|cccc|ccc} & 3 & 5 & 6 & 7 & 4 & 2 & 1 \\ \hline 3 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 5 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 6 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 7 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right),$$

where the row and column numbers have been written in for reference. We'll call this a *Hamming matrix*.

The corresponding H-matrix is

$$H = \left(\begin{array}{ccc} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right).$$

If we feed a length-7 word c to H, it yields a 3-bit vector (a_4, a_2, a_1) where $a_1 = 0$ if the column-1 parity check is verified, and $a_1 = 1$ if that check fails, and similarly for a_2 and a_4. This is because $(c_3, c_5, c_6, c_7, c_1, c_2, c_4)H = (c_5 + c_6 + c_7 + c_4, c_3 + c_6 + c_7 + c_2, c_3 + c_5 + c_6 + c_1)$.

Looking back on how the special Hamming check-bits were constructed, $a_1 = 1$ means that an error occurred in one of the bits of c whose position number ends with 1; $a_1 = 0$ means that if an error occurred it was in one of the other bits, those whose position number ends in 0. (We assume throughout here that *at most a single error occurred*). So a_1 tells us what the last bit of the position number of the error is.

Similarly, $a_2 = 1$ means that an error occurred in one of the bits whose position number has middle bit 1, and $a_2 = 0$ means that if an error occurred it was in one of the other bits, those whose position number has middle bit 0. So a_2 is equal to the middle bit of the position number of the error.

Finally, in exactly the same way, a_4 is equal to the leading bit of the position number of the error.

Put another way, the vector (a_4, a_2, a_1), read as a binary number $a_4a_2a_1$, gives the location of the error. (If all three are 0, there was no error).

3. Examples.

With the 4×7 Hamming-code matrix G, we encode $(1, 0, 0, 1)$ as $c = (1, 0, 0, 1, 1, 0, 0)$. Suppose an error was made in the third bit, i.e. in position 6. So the transmitted word would be $c' = (1, 0, 1, 1, 1, 0, 0)$. Feeding c' to H yields $c'H = (1, 1, 0)$ and the binary number $110 = 6$. So c' can be corrected to $(1, 0, 0, 1, 1, 0, 0)$, and the word correctly decoded as $(1, 0, 0, 1)$.

Another example with the same matrix: we encode $(0, 1, 0, 1)$ as $(0, 1, 0, 1, 0, 1, 0)$. Suppose an error is made in the fifth bit, i.e. in position 4 (this is a check bit, but they also are subject to errors in transmission), so that the transmitted word is $c' = (0, 1, 0, 1, 1, 1, 0)$. Applying H to this c' yields $c'H = (1, 0, 0)$ and the binary number $100 = 4$. So c' can be corrected to $(0, 1, 0, 1, 0, 1, 0)$, and the word correctly decoded as $(0, 1, 0, 1)$.

Anthony Phillips
Revised, 12/8/10.