Math 259 - Spring 2019
Homework 4 Solutions

1. (a) The Hamming distance measures in how many entries two codewords are different. By definition, this number must be nonnegative. In addition, two codewords differ in zero entries if and only if they are the same codeword.

   (b) c_1 differs from c_2 in as many entries as c_2 differs from c_1; in other words these two values are equal since they count the number of entries where c_1 and c_2 are different.

   (c) Fix two codewords c_1 and c_2, and an entry where they differ, say the ith entry. Now consider a third codeword c_3. There are exactly three things that can happen:

   - Maybe the pairs c_1 and c_3 and c_2 and c_3 are both different in that entry. In that case the ith entry contributes to both the count $d(c_1, c_3)$ and to the count $d(c_3, c_2)$.
   - Maybe c_1 and c_3 differ in that entry, but c_2 and c_3 are the same in that entry. In that case the ith entry contributes to the count $d(c_1, c_3)$ but not to the count $d(c_3, c_2)$.
   - Maybe c_1 and c_3 are the same in that entry, but c_2 and c_3 are different in that entry. In that case the ith entry does not contribute to the count $d(c_1, c_3)$ but does contribute to the count $d(c_3, c_2)$.

   Note that it is impossible for both c_1 and c_3 and c_3 and c_2 to be the same in that entry as that would imply that c_1 and c_2 are the same in that entry. But we assumed that we picked an entry where c_1 and c_2 differ.

   From this analysis, we see that whenever an entry contributes a count of 1 to $d(c_1, c_2)$, which is the left hand side of the inequality (because c_1 and c_2 differ in that entry), then the same entry contributes either a count of 1 to both $d(c_1, c_3)$ and $d(c_3, c_2)$, or a count of 1 only to $d(c_1, c_3)$, or a count of 1 only to $d(c_3, c_2)$. In any case, that entry contributes at least a count of 1 to the right hand side of the inequality. Therefore we conclude that $d(c_1, c_2) \leq d(c_1, c_3) + d(c_3, c_2)$.

   (Note that of course, it is possible for c_1 and c_2 to be *the same* at an entry, but different from c_3 in that entry, which would make the right hand side even larger. But for the purposes of our proof, we need not even consider these entries.)

2. (a) Suppose that $d(C) \geq s + 1$. Now if a codeword c is sent, and s or fewer errors are made, then the received message v will satisfy $d(c, v) \leq s$. Since $d(C) \leq d(c, c')$ for any codeword c', the fact that $d(c, v) \leq s < d(C)$ implies that v is not a codeword. Therefore when we receive this message, we know that an error was made.

   (b) Suppose that $d(C) \geq 2t + 1$. Now if a codeword c is sent and t or fewer errors are made, then the received message v will satisfy $d(c, t) \leq t$. Let c' be any codeword other than c. Then by the triangle inequality we have

$$d(c, c') \leq d(c, v) + d(v, c').$$

The left hand side satisfies

$$2t + 1 \leq d(C) \leq d(c, c'),$$

by hypothesis, and the right hand side satisfies

$$d(c, v) + d(v, c') \leq t + d(v, c').$$

Bringing these together, we have

$$2t + 1 \leq d(C) \leq d(c, v) + d(v, c') \leq t + d(v, c').$$

or simply

$$2t + 1 \leq t + d(v, c').$$

From this it follows that $d(v, c') \geq t + 1$. Recall that this is true for any codeword c' that is not c. Therefore, the nearest neighbor to v is c (since $d(c, v) \leq t$ and $d(v, c') \geq t + 1$ for any other c'), and v successfully decodes to c by the nearest neighbor algorithm. Therefore the t errors are corrected.

3. (a) There are many ways to show this. The quickest is to say that $(0, 0, 0)$ does not belong to C. (A linear code is a linear subspace and a subspace always contains 0.) Perhaps the next quickest is to show that the code is not closed under addition: For example $(0, 0, 1) + (1, 1, 1) = (1, 1, 0)$ does not belong to C. In fact it looks like maybe none of the sums of two codewords belong to C? So in any case there are many ways in which C fails to be linear.

   (b) Here we **cannot** use the minimum weight trick because the code is not linear. Therefore sadly we must just compute the distances between any two pairs of codewords and choose the smallest one. If we have

$$c_1 = (0, 0, 1), \quad c_2 = (1, 1, 1), \quad c_3 = (1, 0, 0), \quad \text{and} \quad c_4 = (0, 1, 0),$$

then we have

$$\begin{aligned} d(c_1, c_2) = 2, \quad d(c_1, c_3) = 2, \quad d(c_1, c_4) = 2, \quad d(c_2, c_3) = 2, \\ d(c_2, c_4) = 2, \quad d(c_3, c_4) = 2. \end{aligned}$$

(If you are surprised by this, imagine that these are the coordinates of the vertices of a $1 \times 1 \times 1$ cube embedded in $\mathbb{R}^3$. Then the Hamming distance is just the distance between the vertices if we can only walk along the edges of the cube (no cutting through the cube) and here they have picked all of the four vertices that as "as far apart from each other as possible," which is a distance of 2 if we want to be able to pick four vertices.) Anyway, $d(C) = 2$.

4. First, since C is linear we have that $0 \in C$, so for any $0 \neq c \in C$, we have

$$\mathrm{wt}(c) = d(0, c) \geq d(C).$$

From this it follows that

$$d(C) \leq \min\{\mathrm{wt}(c) : 0 \neq c \in C\}.$$

For the other direction, let c_1 and c_2 be two different codewords. Then $0 \neq c = c_1 - c_2 \in C$, since C is linear. We claim that

$$\mathrm{wt}(c) = d(c_1, c_2).$$

Indeed, $\mathrm{wt}(c)$ counts the number of entries where c is nonzero. But these are exactly the entries where c_1 and c_2 differ, and therefore this is exactly $d(c_1, c_2)$.

Therefore we have that for any pair $c_1 \neq c_2$,

$$d(c_1, c_2) = \mathrm{wt}(c) \geq \min\{\mathrm{wt}(c) : 0 \neq c \in C\}.$$

From this it follows that

$$d(C) \geq \min\{\mathrm{wt}(c) : 0 \neq c \in C\}.$$

Since we have both $d(C) \geq \min\{\mathrm{wt}(c) : 0 \neq c \in C\}$ and $d(C) \leq \min\{\mathrm{wt}(c) : 0 \neq c \in C\}$, it follows that

$$d(C) = \min\{\mathrm{wt}(c) : 0 \neq c \in C\}.$$

5. (a) The generating matrix generates a code in the following way: If G is a $k \times n$ matrix, take all possible vectors of length k and multiply them by G on the right. This will give all vectors of C, and they will have length n. Here G is a 2×4 matrix, so it will generate codewords of length 4 when we start with vectors of length 2. The codewords are

$$\begin{aligned} c_1 &= (0,0)G = (0,0,0,0) \\ c_2 &= (1,0)G = (1,0,1,1) \\ c_3 &= (0,1)G = (0,1,0,1) \\ c_4 &= (1,1)G = (1,1,1,0). \end{aligned}$$

(b) We have $n = 4$ (the length of the code) and $k = 2$ (the dimension of the code).

(c) Here

$$P = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix},$$

so

$$H = \begin{pmatrix} -P^T & I \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

(d) Since this code **is** linear, we can use the minimum weight of a nonzero codeword to compute the minimum distance. We have $\mathrm{wt}(c_2) = 3$, $\mathrm{wt}(c_3) = 2$ and $\mathrm{wt}(c_4) = 3$, so the minimum weight of a nonzero codeword is 2 and $d(C) = 2$.

(e) We have that C can detect s errors for $d(C) \geq s + 1$. Here $2 \geq s + 1$ means that $1 \geq s$, so this code can detect at most one error.

We have that C can correct t errors for $d(C) \geq 2t + 1$. Here $2 \geq 2t + 1$ means that $\frac{1}{2} \geq t$, so this code cannot correct any errors.

6. (a) i. To prove that a subset of a vector space is a subspace, it suffices to show that it is closed under linear combinations. Therefore, it suffices to show that for any a_1 and a_2 in $\mathbb{F}_2$, the linear combination

$$a_1(0, 0, 0) + a_2(1, 1, 1) = a_2(1, 1, 1)$$

belongs to C_3. Since a_2 can only be 0 or 1, $a_2(1, 1, 1)$ is either $(0, 0, 0)$ or $(1, 1, 1)$, and the result follows.

ii. Here $n = 3$ and $k = 1$.

iii. Since this code is linear we can use our minimum weight result. Since there is only one nonzero codeword, we have that $d(C_3) = \mathrm{wt}((1, 1, 1)) = 3$.

iv. We solve $d(C_3) = 3 \geq 2t + 1$ and get $1 \geq t$, so C_3 can correct at most 1 error.

(b) i. Again it suffices to show that C_4 is closed under linear combinations. This time, it means that we must show that for any a_1 and a_2 in $\mathbb{F}_2$, the linear combination

$$a_1(0, 0, 0, 0) + a_2(1, 1, 1.1) = a_2(1, 1, 1, 1)$$

belongs to C_4. Again, since a_2 can only be 0 or 1, $a_2(1, 1, 1, 1)$ is either $(0, 0, 0, 0)$ or $(1, 1, 1, 1)$, and the result follows.

ii. Here $n = 4$, but $k = 1$ still.

iii. Again this code is linear so we can use our minimum weight result. Again there is only one nonzero codeword, we have that $d(C_4) = \mathrm{wt}((1, 1, 1, 1)) = 4$.

iv. We solve $d(C_4) = 4 \geq 2t + 1$ and get $\frac{3}{2} \geq t$, so C_4 can also correct at most 1 error.

(c) i. We show that C_n is closed under linear combinations. Indeed, for any a_1 and a_2 in $\mathbb{F}_2$, the linear combination

$$a_1(0, 0, \ldots, 0) + a_2(1, 1, \ldots, 1) = a_2(1, 1, \ldots, 1)$$

belongs to C_n, because a_2 can only be 0 or 1, so $a_2(1, 1, \ldots, 1)$ is either $(0, 0, \ldots, 0)$ or $(1, 1, \ldots, 1)$, and the result follows.

ii. Here $n = n$ (phew, no confusion there) but $k = 1$ still.

iii. We use our minimum weight result, which says that $d(C_n) = \mathrm{wt}((1, 1, \ldots, 1)) = n$.

iv. We solve $d(C_n) = n \geq 2t + 1$ and get $\frac{n-1}{2} \geq t$, so C_n can correct at most $\lfloor \frac{n-1}{2} \rfloor$ errors.

7. (a) A parity check matrix for this code is

$$H = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

   (b) i. The syndrome is $[0, 1, 0, 0, 1, 1, 1, 1]$, this is not a codeword.
   ii. The syndrome is $[0, 0, 0, 0, 0, 0, 0, 0]$, this is a codeword.
   iii. The syndrome is $[0, 0, 0, 0, 0, 0, 0, 0]$, this is a codeword.
   iv. The syndrome is $[0, 1, 1, 1, 0, 0, 1, 1]$, this is not a codeword.

8. (a) For this received message, the syndrome $v_1 H^T$ is $[1, 0, 1]$, which is the second column of H. Therefore we flip the second bit of v_1 to obtain the codeword $c_1 = [1, 0, 1, 0, 1, 0, 1]$.

   (b) This time, the syndrome $v_2 H^T$ is $[1, 0, 0]$, which is the fifth column of H. Therefore we flip the fifth bit of v_2 to obtain the codeword $c_2 = [1, 0, 1, 0, 1, 0, 1]$.

   (c) The syndrome $v_3 H^T$ is $[0, 0, 0]$ so in fact v_3 is a codeword and does not need to be corrected.

   (d) Finally, the syndrome $v_4 H^T$ is $[1, 1, 0]$. This is the first column of H, so we flip the first bit of v_4 to obtain the codeword $c_4 = [0, 0, 1, 0, 0, 1, 1]$.

9. There actually was an error in the statement of the problem. The minimum distance of this code is 5, which means it can correct 2 errors, not 5 errors. If you put in more than 2 errors in fact your message should not decrypt.

10. (a) The public key would be the "scrambled" generating matrix denoted G_1 in the book, which is

$$G_1 = SGP = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix},$$

   as well as the information $t = 1$, to let people know that the code that you are using can correct 1 error.

   (b) In each case, given a ciphertext y, the decryption algorithm will go through the following steps, because you are using a Hamming code:

- Compute $y_1 = yP^{-1}$. This is a codewords plus an error, so we must decode this vector.
- Compute the syndrome $y_1 H^T$.
- This will give a column of H, we change the corresponding bit in y_1 to obtain the decoded vector x_1.
- Get the "premessage" x_0 by reading only the first 4 bits of x_1. (In other words, remove the extra check bits from x_1 to get x_0.
- Compute the message $x = x_0 S^{-1}$.

We see that this requires us to compute P^{-1}, H (the parity check matrix), and S^{-1}, which we can do with the secret key that we have. We have:

$$P^{-1} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad H = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

and

$$S^{-1} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}.$$

Thus we decrypt!

i. We start with $y = [0, 0, 1, 0, 0, 1, 0]$.
   - We have $y_1 = [1, 0, 0, 0, 0, 0, 1]$
   - The syndrome is $y_1 H^T = [1, 1, 1]$.
   - This is the fourth column of H, so we flip the fourth bit of y_1 to get the decoded vector $x_1 = [1, 0, 0, 1, 0, 0, 1]$
   - The "premessage" x_0 is just then $[1, 0, 0, 1]$, the first four bits.
   - The actual message was $x = x_0 S^{-1} = [1, 0, 0, 0]$.

ii. This time we start with $y = [1, 0, 1, 0, 0, 1, 1]$.
   - We have $y_1 = yP^{-1} = [1, 1, 0, 1, 0, 0, 1]$
   - The syndrome is $y_1 H^T = [1, 0, 1]$.
   - This is the second column of H, so we flip the second bit of y_1 to get the decoded vector $x_1 = [1, 0, 0, 1, 0, 0, 1]$. This is the same as just before!
   - The "premessage" x_0 is again then $[1, 0, 0, 1]$, the first four bits.
   - The actual message was $x = x_0 S^{-1} = [1, 0, 0, 0]$. So we see that because of the random error, the same message doesn't even get encrypted to the same ciphertext twice!

iii. Now we start with $y = [0, 0, 1, 1, 1, 0, 1]$.
- We have $y_1 = yP^{-1} = [1, 1, 1, 0, 0, 1, 0]$
- The syndrome is $y_1 H^T = [0, 1, 0]$.
- This is the sixth column of H, so we flip the sixth bit of y_1 to get the decoded vector $x_1 = [1, 1, 1, 0, 0, 0, 0]$.
- The "premessage" x_0 is again then $[1, 1, 1, 0]$, the first four bits.
- The actual message was $x = x_0 S^{-1} = [1, 1, 0, 0]$.

iv. Ok, one last one, I think we're getting the hang of this. We start with $y = [0, 1, 1, 1, 1, 0, 0]$.
- We have $y_1 = yP^{-1} = [1, 0, 1, 0, 1, 1, 0]$
- The syndrome is $y_1 H^T = [0, 1, 1]$.
- This is the third column of H, so we flip the third bit of y_1 to get the decoded vector $x_1 = [1, 0, 0, 0, 1, 1, 0]$.
- The "premessage" x_0 is again then $[1, 0, 0, 0]$, the first four bits.
- The actual message was $x = x_0 S^{-1} = [0, 1, 1, 1]$.