Coding Theory Homework

Week 8 (Section 2.7 - 2.8)

Exercise 2.7.4 (b)

> Find a parity-check matrix from each of the following codes.

> b. C = {0000, 1001, 0110, 1111}

In exercise 2.6.5 (b) of week 7 we found that the generator matrix for C is:

```
┌ 1001 ┐
└ 0110 ┘
```

By algorithm 2.5.7, we construct H:

```
┌ 1001 ┐
└ 0110 ┘  = [I X]


    ┌ X ┐    ┌ 01 ┐
H = └ I ┘  = │ 10 │
             │ 10 │
             └ 01 ┘
```

Exercise 2.7.9 (b)

> In each part, a parity-check matrix for linear code C is given. Find (i) a generator matrix for $C^{\perp}$; (ii) a generator matrix for C.

> b.

Matrix H_C

```
┌ 01 ┐
│ 10 │
│ 01 │
```

```
│ 10 │
└ 01 ┘
```

By transposing H_C we get the generator matrix for $C^\perp$: $G_{C^\perp}$

```
┌ 01010 ┐
└ 10101 ┘
```

By applying algorithm 2.5.7 we get the parity-check matrix for $C^\perp$: $H_{C^\perp}$

```
┌ 01010 ┐
└ 10101 ┘

-- Swap row 1 with row 2

┌ 10101 ┐
└ 01010 ┘

-- RREF

┌ 10 ┆ 101 ┐
└ 01 ┆ 010 ┘

-- Matrix X

┌ 101 ┐
└ 010 ┘

-- Matrix H

┌ 101 ┐
│ 010 │
│ 100 │
└ 010 ┘
```

Transposing the parity-check matrix $H_{C^\perp}$ gives the generator matrix for C: G_C

```
┌ 1010 ┐
│ 0101 │
└ 1000 ┘
```

Exercise 2.7.10

> List all the words of the dual code $C^{\perp}$ for the code C = {00000, 11111}. Then find the generating and parity-check matrices for $C^{\perp}$.

$C^{\perp}$ = {00000}

Exercise 2.7.11 (b)

> For each code C described below, find the dimension of C, the dimension of $C^{\perp}$, the size of the generating and parity-check matrices for C and for $C^{\perp}$, the number of words in C and in $C^{\perp}$, and the information rates r or C and $C^{\perp}$.

> b. C has length n = 23 and dimension 11.

Exercise 2.8.4 (b)

> Let C be the generator matrix in Example 2.8.3. Encode each of the following messages u, and observe that the first 4 digits in the resulting codeword form the message u.

```
┌ 1000101 ┐
| 0100100 |
| 0010110 |
└ 0001011 ┘
```

> a. u = 1111

```
1000101 · 1
0100100 · 1
0010110 · 1
0001011 · 1
──────────── +
1111100
```

> b. u = 1011

```
1000101 · 1
0100100 · 0
0010110 · 1
0001011 · 1
──────────── +
1011000
```

> c. u = 0000

```
1000101 · 0
0100100 · 0
0010110 · 0
0001011 · 0
——————————— +
0000000
```

Exercise 2.8.5

> Explain a method for recovering u from uG if G is not in standard form.

Take for example the matrix G as used in exercise 2.6.11 which is not in standard form.

```
┌ 10110 ┐
│ 01011 │
└ 00101 ┘
```

For all words in K^3 we will determine the uG:

```
(000)    0·10110 + 0·01011 + 0·00101 = 00000
(100)    1·10110 + 0·01011 + 0·00101 = 10110
(010)    0·10110 + 1·01011 + 0·00101 = 01011
(001)    0·10110 + 0·01011 + 1·00101 = 00101
(110)    1·10110 + 1·01011 + 0·00101 = 11101
(101)    1·10110 + 0·01011 + 1·00101 = 10011
(011)    0·10110 + 1·01011 + 1·00101 = 01110
(111)    1·10110 + 1·01011 + 1·00101 = 11000
```

If we take the first 3 digits of each word in uG and multiply this with the original matrix G

```
(000)    0·10110 + 0·01011 + 0·00101 = 00000   ->   (000)
(101)    1·10110 + 0·01011 + 1·00101 = 10011   ->   (100)
(010)    0·10110 + 1·01011 + 0·00101 = 01011   ->   (010)
(001)    0·10110 + 0·01011 + 1·00101 = 00101   ->   (001)
(111)    1·10110 + 1·01011 + 1·00101 = 11000   ->   (110)
(100)    1·10110 + 0·01011 + 0·00101 = 10110   ->   (101)
(011)    0·10110 + 1·01011 + 1·00101 = 01110   ->   (011)
```

```
(110)    1·10110 + 1·01011 + 0·00101 = 11101   ->   (111)
```

The first 3 digits of the resulting words correspond with the original *u*.

Exercise 2.8.6

> If a linear code *C* has the following generator matrix, recover *u* from *v* = *uG* = 0000101

```
┌ 1100101 ┐
│ 0110101 │
│ 1011011 │
│ 1100110 │
└ 0110000 ┘
```

Using the first 5 digits of `0000101` ie `00001` and mutliplying this with *G* we get:

```
0·1100101 + 0·0110101 + 0·1011011 + 0·1100110 + 1·0110000 = 0110000
```

The first 5 digits of the resulting `0110000` ie `01100` form the original *u*.

Exercise 2.8.10 (b)

> Find a systematic code *C′* equivalent to the given code *C*. Check that *C* and *C′* have the same length, dimension, and distance.

> b. *C* = {00000, 11100, 00111, 11011}

columns: [a, b, c, d, e] -> [a, d, e, b, c]

C = {00000, 10011, 01101, 11110}

Exercise 2.8.11 (b)

> Find a generator matrix *G* in standard form for a code equivalent to the code with given matrix *G*

> b.

```
┌ 111000000 ┐
```

```
│ 000111000 │
└ 000111111 ┘
```

```
┌ 111000000 ┐
│ 000111000 │
└ 000111111 ┘

-- Add row 2 to row 3

┌ 111000000 ┐
│ 000111000 │
└ 000000111 ┘

-- RREF
-- Swap column 4 with column 2

┌ 101100000 ┐
│ 010011000 │
└ 000000111 ┘

-- Swap column 7 with column 3

┌ 100110000 ┐
│ 010001100 │
└ 001000011 ┘
```

Exercise 2.8.12 (b)

> Find a generator matrix G' in standard form for a code C' equivalent to the code C with given parity-check matrix H

> b.

```
┌ 100 ┐
│ 111 │
│ 010 │
│ 110 │
│ 101 │
│ 001 │
└ 011 ┘
```

Exercise 2.8.13

Exercise 2.8.14