# Error correlation - Lab 6

By Nikan Mahdavi Tabatabaei, AU-ID: au716549 & Jesper Bertelsen: AU-ID: au689481

Indholdsfortegnelse

[Error correlation - Lab 6 1](#_Toc166779262)

[Task 1. 1](#_Toc166779263)

[Task 2. n = ulige / lige, stor eller lille, hvad er bedst? 2](#_Toc166779264)

[Task 3. Lav egen encoder og decoder som kan lave korrelation. 2](#_Toc166779265)

[Task 4 3](#_Toc166779266)

[Task 5 - Varying n 5](#_Toc166779267)

[Task 6 7](#_Toc166779268)

[Task 7 8](#_Toc166779269)

[Task 8 9](#_Toc166779270)

[Task 9 10](#_Toc166779271)

[Task 10 11](#_Toc166779272)

[Conclusion 11](#_Toc166779273)

## Task 1.

Repetition Code:

Every repetition of length *n*.

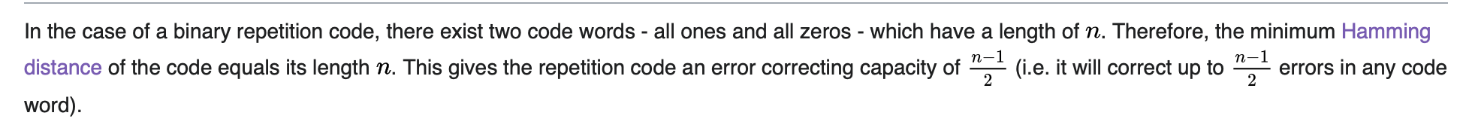
Two Codes:

Hamming distance:

The amount of repetitions that are required to convert one array to other.

If one is filled with ones and the other with zeroes, then it requires the lists length in order to change all.

Thus the hamming distance is n, as we know.



*Wikipedia: repetition code*

Therefore our error correcting capacity is

## Task 2. *n = ulige / lige, stor eller lille, hvad er bedst?*

The bigger n is, the bigger is the probability, which is quite desirable.

Since the probability function rounds down, then will *n* *= odd* mean we have a bigger probability.

=========

=========

## Task 3. Lav egen encoder og decoder som kan lave korrelation.

In this task and the next task we will be using the repetition coding.

For a message bit:

The repetitionbits will be of

For *n* being the amount of times the bits is repeated.

For majority voting in the decoder, the maximum amounts of errors can be the half the size of *n* - 1 before the errors become the majority of the bits.

The maximum amounts of flipped bits :

For task 3 and 4 we made the system as such:

Source bits

Channel encoder

Modulator

A

W

G

N



Demodulator

Channel decoder

bits

Which will be explained in task 4.

## Task 4

The blocks has been build up like so:

The Channel encoder:

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype

Automatisk genereret beskrivelse

It’s function is to take the bits, then make a new array, where each bit is repeated *n* times.

The Modulator:

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype

Automatisk genereret beskrivelse

We chose to use the I, Q modulator for this.

We chose a threshold of 0,8 making

With *r* being the length of the complex coefficients made up of .

The AWGN:

The Add White Gaussian Noise is a matlab function adding noice to the modulated signal depending on the SNR variable, which we vary in later plots.

The Demodulator:

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype

Automatisk genereret beskrivelse

Nothing major here. When the *Y* exceeds the threshold of 0,8, then its translated to the bit value 1, else 0.

The Channel Decoder:

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype, nummer/tal

Automatisk genereret beskrivelse

In the channel decoder we shrink the data set. Depended on the value of *n*, we check for the amounts of ones and the amounts of zeros. When the amounts of repeated bits have been checked, we do a majority vote, adding 1 to the new bits if the repetition bits consist of more 1’s than zeros, else 0 is added.

The bits returned is our final codeword.

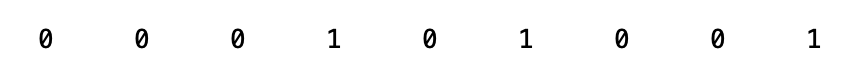
## Task 5 - Varying *n*

For this task we set the SNR constant as 3;

Our codeword as input is:

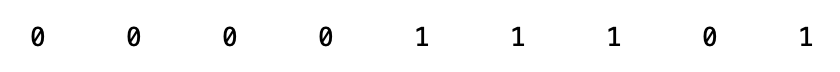
Having a length of 9.

With n = 3:



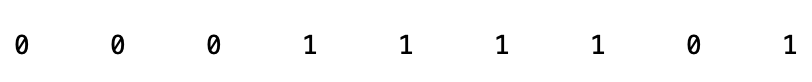
With errors at index, 5 & 7.

With n = 5:



With an error at 4.

With n = 7:

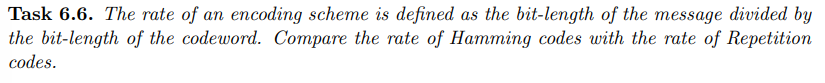


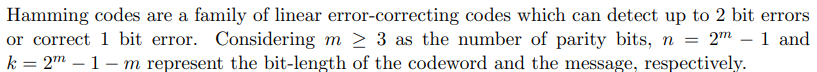
With no errors.

Adding more repetitions ensures that the majority vote has a better chance to make the right decision.

That being said, the Gaussian Noise wasn’t constant doing this, so the results could have looked different. But enlarging *n* means less probability of errors.

## Task 6





For Hamming Code:

For Repetition Code:

A screenshot of a computer screen

Description automatically generated

The most normal values are:

We will get:

This means that the rate of repetition code is larger than the rate of hamming code.

## Task 7

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

This makes great sense. Because since for hamming code, the error correcting capability of hamming code will be , meaning that more errors than one in the code cannot be corrected properly, which we do by adding one more error to the code and see if the number of errors increases at the output (is not 0 anymore). But here all corruptions in the encoded massage were detected.

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

Just as seen above our theory is correct.

We know for hamming code that:

This means that increasing m increases k (if n is held constant) and hence the rate and thus increases the number of errors and thus lower probability of correction although a smaller bandwidth usage.

## Task 8

Our implementation of the hamming code.Et billede, der indeholder tekst, skærmbillede, Font/skrifttype

Automatisk genereret beskrivelse

## Task 9

Making plots for bit error to SNR:

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype, nummer/tal

Automatisk genereret beskrivelse

Et billede, der indeholder tekst, skærmbillede, Font/skrifttype, linje/række

Automatisk genereret beskrivelse

Just code combining both repetitioncode and hamming code.

Et billede, der indeholder linje/række, Kurve, diagram, skærmbillede

Automatisk genereret beskrivelseSimulating

It seems as the biterror with repetitioncode is generally falling, but the hamming code doesn’t seem to change.

By theory we know, that the hamming code should have the lowest bit error, so it might just be something within the matlab code, that doesn’t function right.

## Task 10

We tried the given conditions on task 9 and the results and discussion can be seen and read above.

## Conclusion

We know from theory that hamming should be better than repetition code, although according to our simulation Hamming Coding was not as good as its counter part at lower SNRs.