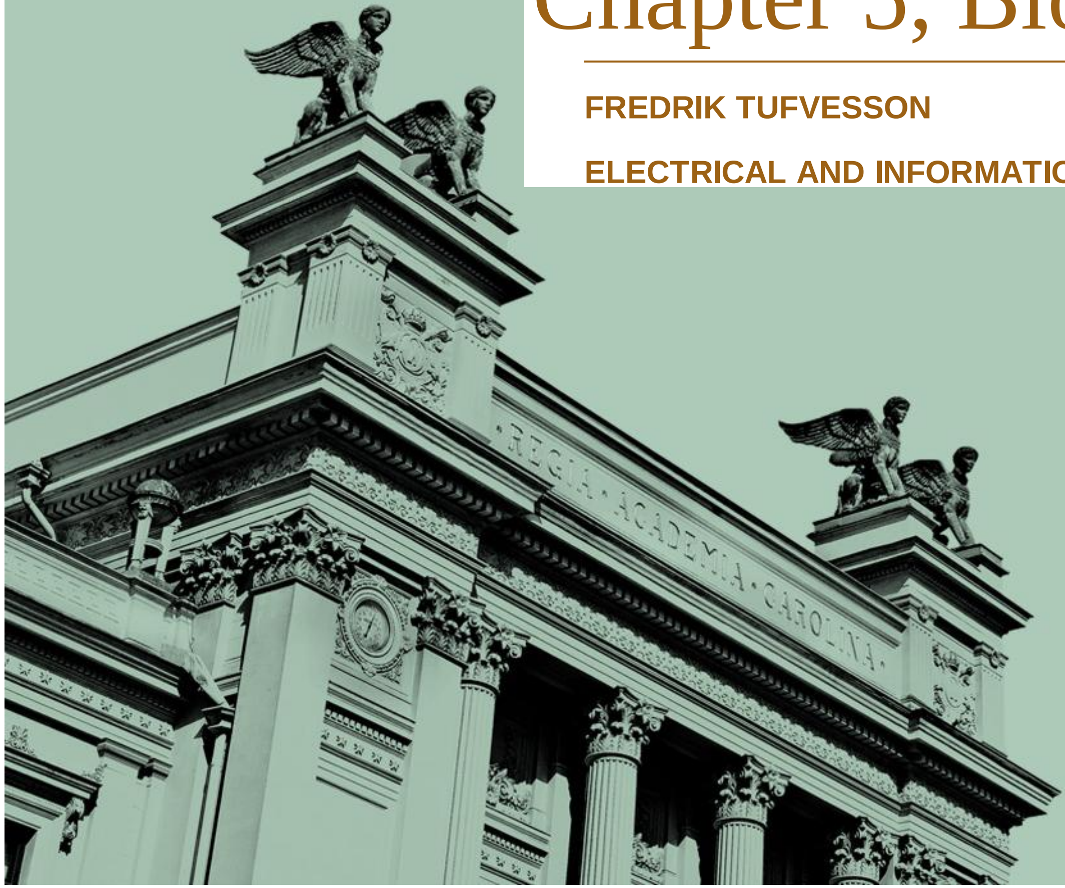
LUND UNIVERSITY

Information Transmission Chapter 5, Block codes

FREDRIK TUFVESSON

ELECTRICAL AND INFORMATION TECHNOLOGY

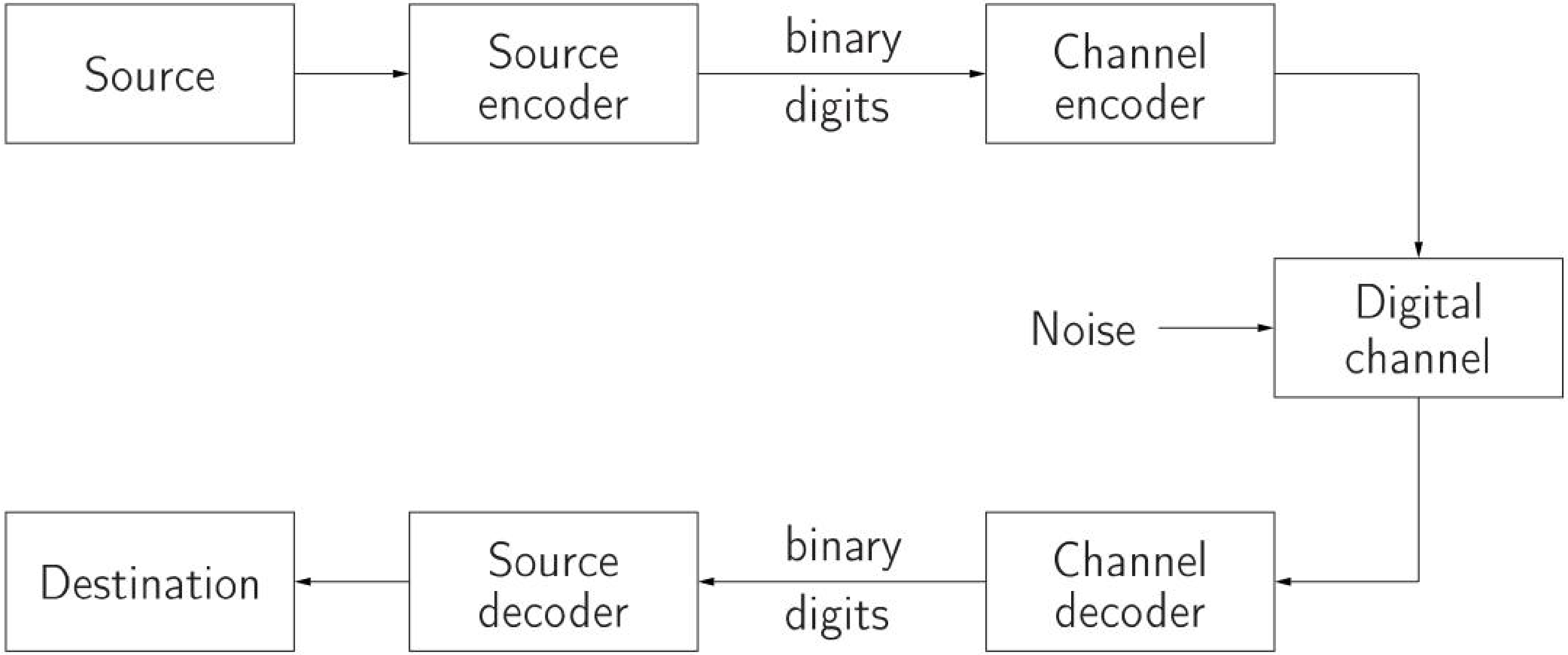


Methods of channel coding

- For channel coding (error correction) we have two main classes of codes, namely:
  - block codes, which we first encountered when we discussed Shannon's channel coding theorem
  - convolutional codes.

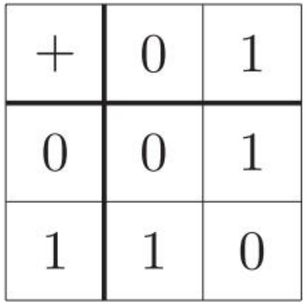
We shall briefly discuss both classes.

A schematic communication system

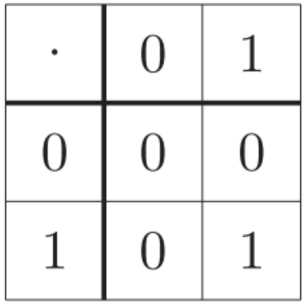


The binary field

For the following calculations we use the *binary field*, for which the rules of addition and multiplication are those of *modulo-two* arithmetic:



$+$	0	1
0	0	1
1	1	0



$\cdot$	0	1
0	0	0
1	0	1

Notice that since 1+1=0, subtraction is the same as addition, which is very convenient

The error pattern

Suppose that the codeword $\mathbf{v} = (v_1\, v_2 \ldots v_N)$ is transmitted over the binary symmetric channel and that $\mathbf{r} = (r_1 r_2 \ldots r_N)$ is the possibly erroneously received version of it, then the *error pattern* $\mathbf{e} = (e_1\, e_2 \ldots e_N)$ is defined to be the *N*-tuple that satisfies

$$\mathbf{r} = \mathbf{v} + \mathbf{e}$$

If we have one error, that is, **e** consists of one 1 and *N-1* 0's, then one component in **v** is altered.

Two errors cause two altered components in **v**.

Minimum distance

The *minimum distance*, d_{min}, of a block code **B** is the minimum of all the distances between two non-identical codewords of the code.

If the sum of any two codewords is a codeword, then the code is said to be *linear*. For a linear block code the minimum distance is simply equal to the least number of 1's in a nonzero codeword

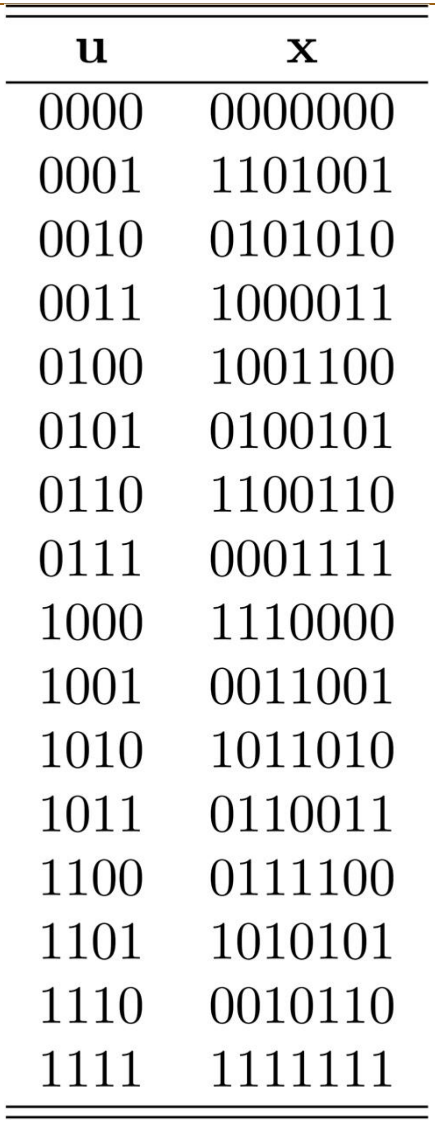
In general, a block code with minimum distance d_{min} will correct up $(d_{min}-1)/2$ errors.

Alternatively, it can be used to *detect* up to $d_{min}-1$ errors.

The (7,4) Hamming code

Hamming constructed a class of *single-error-correcting* linear block codes with minimum distance d_{min}=3.

In the table we specify an encoder mapping for the (7,4) Hamming code with M=2^4=16 codewords.



u	**x**
0000	0000000
0001	1101001
0010	0101010
0011	1000011
0100	1001100
0101	0100101
0110	1100110
0111	0001111
1000	1110000
1001	0011001
1010	1011010
1011	0110011
1100	0111100
1101	1010101
1110	0010110
1111	1111111

Example

Assume that we would like to transmit the information 4-tuple **u**=(1011) over a binary symmetric channel.

Then we encode it, by using the mapping in the table, and obtain the codeword **v**=(0110011).

Let, for example, the sixth position be altered by the channel. Thus, we receive **r**=(0110001).

Example cont

To correct the error we add position-wise modulo-two rows 2, 3, and 7 (the positions corresponding to the 1's in **r**) and obtain

$$\begin{array}{c} 010 \\ 011 \\ 111 \\ \hline 110 \end{array}$$

that is, the binary representation of 6; we flip the sixth position in **r**=(0110001) and obtain the estimate of the codeword $\widehat{\mathbf{v}} = (0110011)$ which corresponds to the information 4-tuple $\widehat{\mathbf{u}} = (1011)$.

How does it work? (I)

Why does our scheme work? We can write the received 7-tuple as the sum of the codeword and the error pattern, that is, **r=v+e.**

Remember that 1+1=0!

Due to this simple equality we can obtain the sum of the rows corresponding to the 1's in **r** by adding component-wise the sums of the rows corresponding to the 1's in **v** and **e**.

How does it work? (II)

Now we exploit that the mapping in the table is constructed such that the sum of the rows corresponding to the 1's in any codeword is 000.

Hence, we conclude that the sum of the rows corresponding to the 1's in **r** (this is the sum that the decoder computes) is equal to the sum of the rows corresponding to the 1's in **e**.

How does it work? (III)

But assuming at most one error during the transmission we obtain in case of no errors the sum of zero rows which we interpret as 000 and then we accept **r** as our estimate$\widehat{\mathbf{v}}$;

In case of one error the sum contains one row, namely, precisely the row which is the binary representation of the position of the 1 in **e**.

Hence, flip that position in **r** and we obtain our estimated codeword $\widehat{\mathbf{v}}$;

The generator matrix

How do we obtain the remarkable encoder mapping?

Since the Hamming code is linear the codewords corresponding to the information 4 tuples 1000, 0100, 0010, 0001 are of particular interest; these codewords form a so-called *generator matrix* for the (7,4) Hamming code:

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

Codeword generation

All codewords can be obtained as the product of the corresponding information 4-tuples and the generator matrix:

$$\mathbf{v} = \mathbf{u}G$$

For example, the codeword corresponding to **u**=(1011) is obtained as the position-wise modulo-two sum of the first, third and fourth rows in *G*, that is,

$$\begin{array}{ccccccc}
1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 \\
\hline
0 & 1 & 1 & 0 & 0 & 1 & 1
\end{array}$$

in agreement with the mapping.

Generation of the parity check matrix

Assume that we have a *K x N* generator *G*, then by the theory of matrices there exists an *(N-K) x N* matrix *H* such that

$$GH^T = \mathbf{0}$$

It follows immediately that

$$\mathbf{u}GH^T = \mathbf{0}$$

that is, we have the fundamental result

$$\mathbf{v}H^T = \mathbf{0}$$

Where H is the so-called parity check matrix.

The parity-check matrix

In words, let **v** be a codeword, then if we add (position-wise modulo-two) the rows of H^T corresponding to the 1s in **v** we obtain the allzero *(N-K)*-tuple.

This computation is a parity-checking procedure and thus we call the matrix *H* a *parity-check matrix* of our code.

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

The generator matrix and the parity check matrix

It is easily verified that

$$GH^T = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} = \mathbf{0}$$

Using linear algebra we can obtain the generator matrix *G* for a given parity-check matrix *H* and vice versa.

Lund
University