Digital Communication Systems
ECS 452

Asst. Prof. Dr. Prapun Suksompong

prapun@siit.tu.ac.th

5. Channel Coding

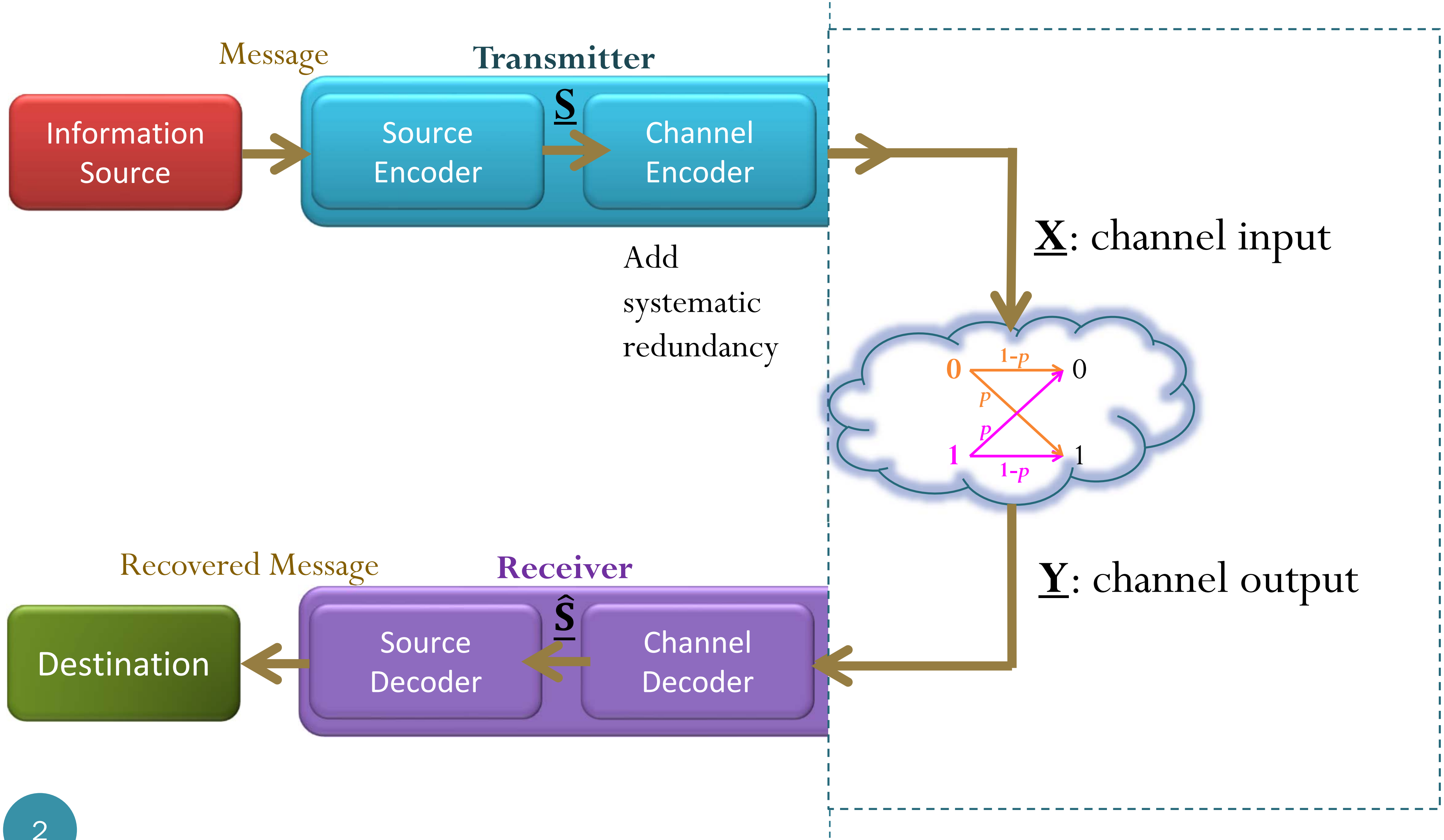
Office Hours:

Check Google Calendar on the course website.

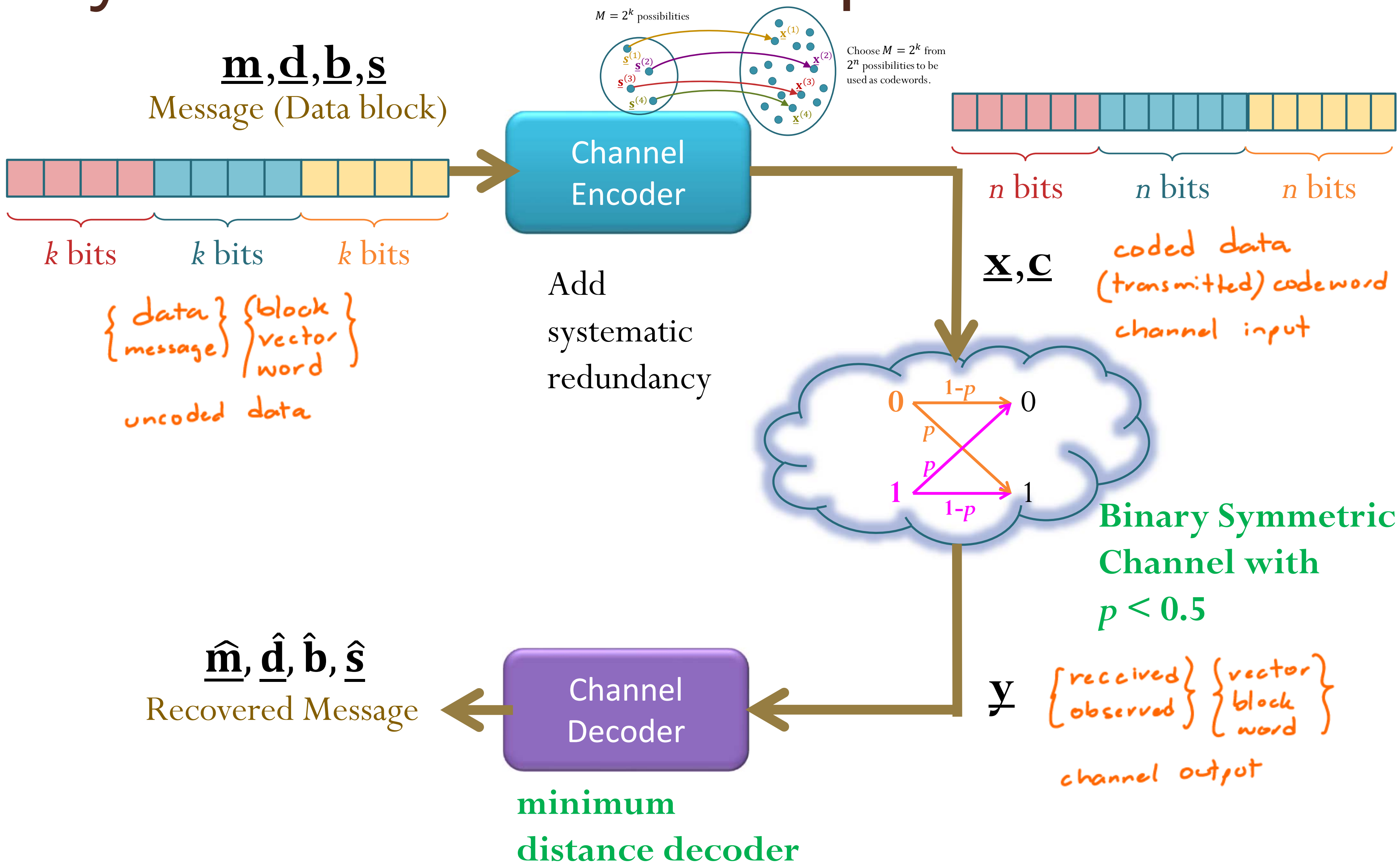
Dr.Prapun's Office:

6th floor of Sirindhralai building, BKD

Review: Channel Encoder and Decoder



System Model for Chapter 5



Vector Notation

- $\vec{\mathbf{v}}$: column vector $\begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_i \\ \vdots \\ v_n \end{pmatrix}$
- $\underline{\mathbf{r}}$: row vector $(r_1, r_2, \ldots, r_i, \ldots r_n)$

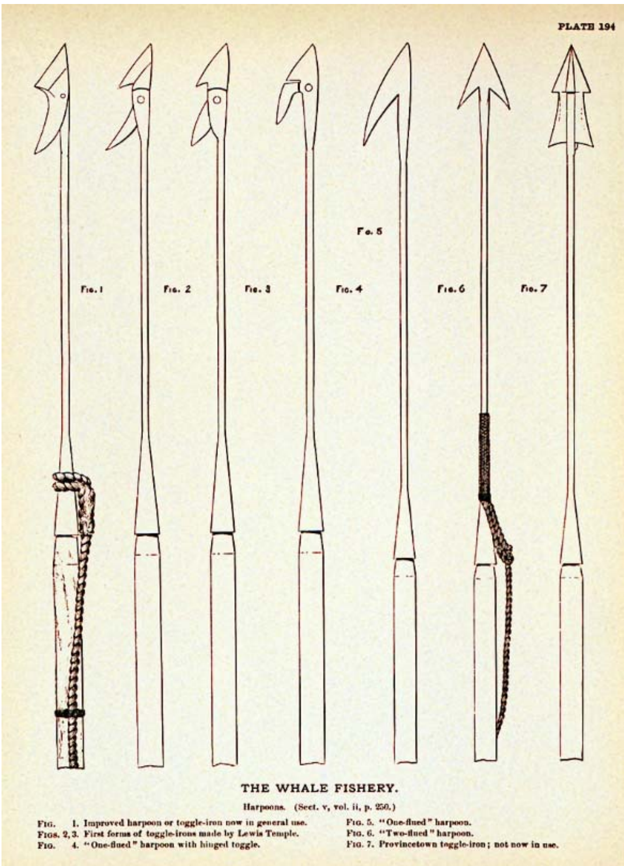
$\vec{0}, \underline{0}$: the zero vector
(the all-zero vector)

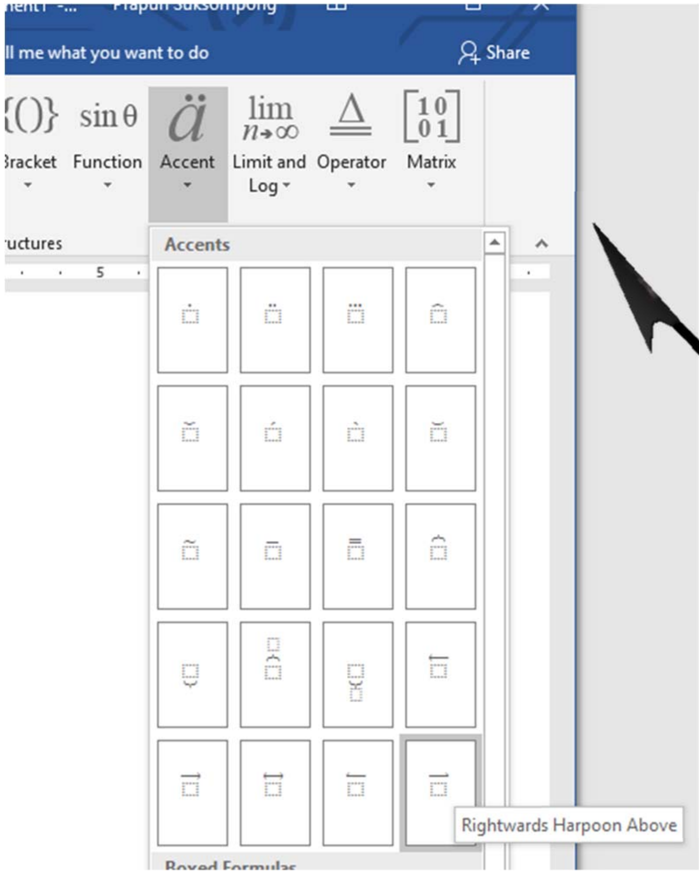
$\vec{1}, \underline{1}$: the one vector
(the all-one vector)

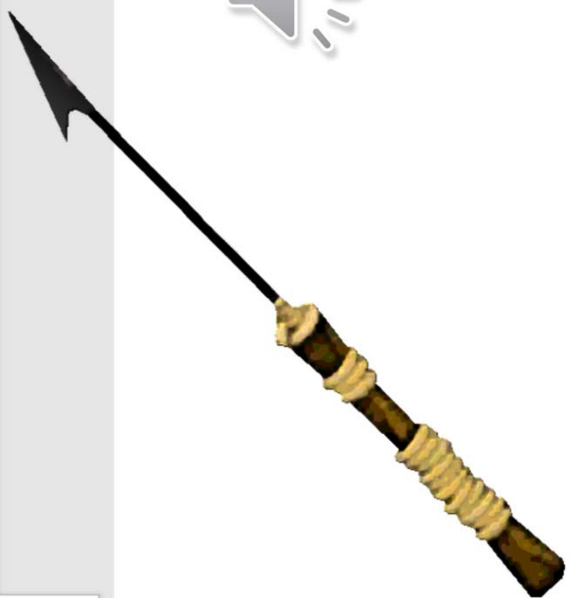
- Subscripts represent element indices inside individual vectors.
  - v_i and r_i refer to the i^{th} elements inside the vectors $\vec{\mathbf{v}}$ and $\underline{\mathbf{r}}$, respectively.
- When we have a list of vectors, we use superscripts in parentheses as indices of vectors.
  - $\vec{\mathbf{v}}^{(1)}, \vec{\mathbf{v}}^{(2)}, \ldots, \vec{\mathbf{v}}^{(M)}$ is a list of M column vectors
  - $\underline{\mathbf{r}}^{(1)}, \underline{\mathbf{r}}^{(2)}, \ldots, \underline{\mathbf{r}}^{(M)}$ is a list of M row vectors
  - $\vec{\mathbf{v}}^{(i)}$ and $\underline{\mathbf{r}}^{(i)}$ refer to the i^{th} vectors in the corresponding lists.

Harpoon

- a long, heavy spear attached to a rope, used for killing large fish or whales





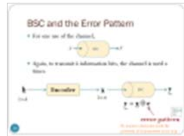


Channel Decoding

- Recall

  MAP decoder is optimal → (Codewords are equally likely) → ML decoder is optimal → (BSC with $p < 0.5$) → Min distance decoder is optimal

  1. **MAP decoder** is the optimal decoder.
  2. When the codewords are equally-likely, the **ML decoder** the same as the MAP decoder; hence it is also **optimal**.
  3. When the **crossover probability** of the BSC p is **< 0.5**,
     ML decoder is the same as the **minimum distance decoder**.



- In this chapter, we assume the use of **minimum distance decoder**.
  - $\hat{\underline{\mathbf{x}}}(\underline{\mathbf{y}}) = \arg\min_{\underline{\mathbf{x}}} d(\underline{\mathbf{x}}, \underline{\mathbf{y}})$
- Also, in this chapter, we will focus
  - less on probabilistic analysis,
  - but more on explicit codes.

Digital Communication Systems ECS 452

Asst. Prof. Dr. Prapun Suksompong

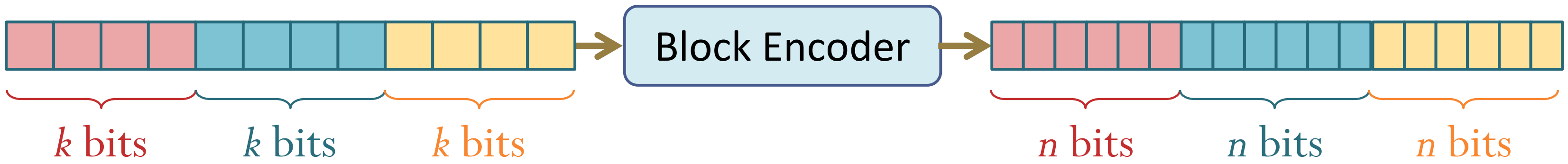
prapun@siit.tu.ac.th

5.1 Binary Linear Block Codes

Two families of codes:
- block (5.1)
- convolutional (5.2)

Review: Block Encoding

- We mentioned the general form of channel coding over BSC.
- In particular, we looked at the general form of **block codes**.

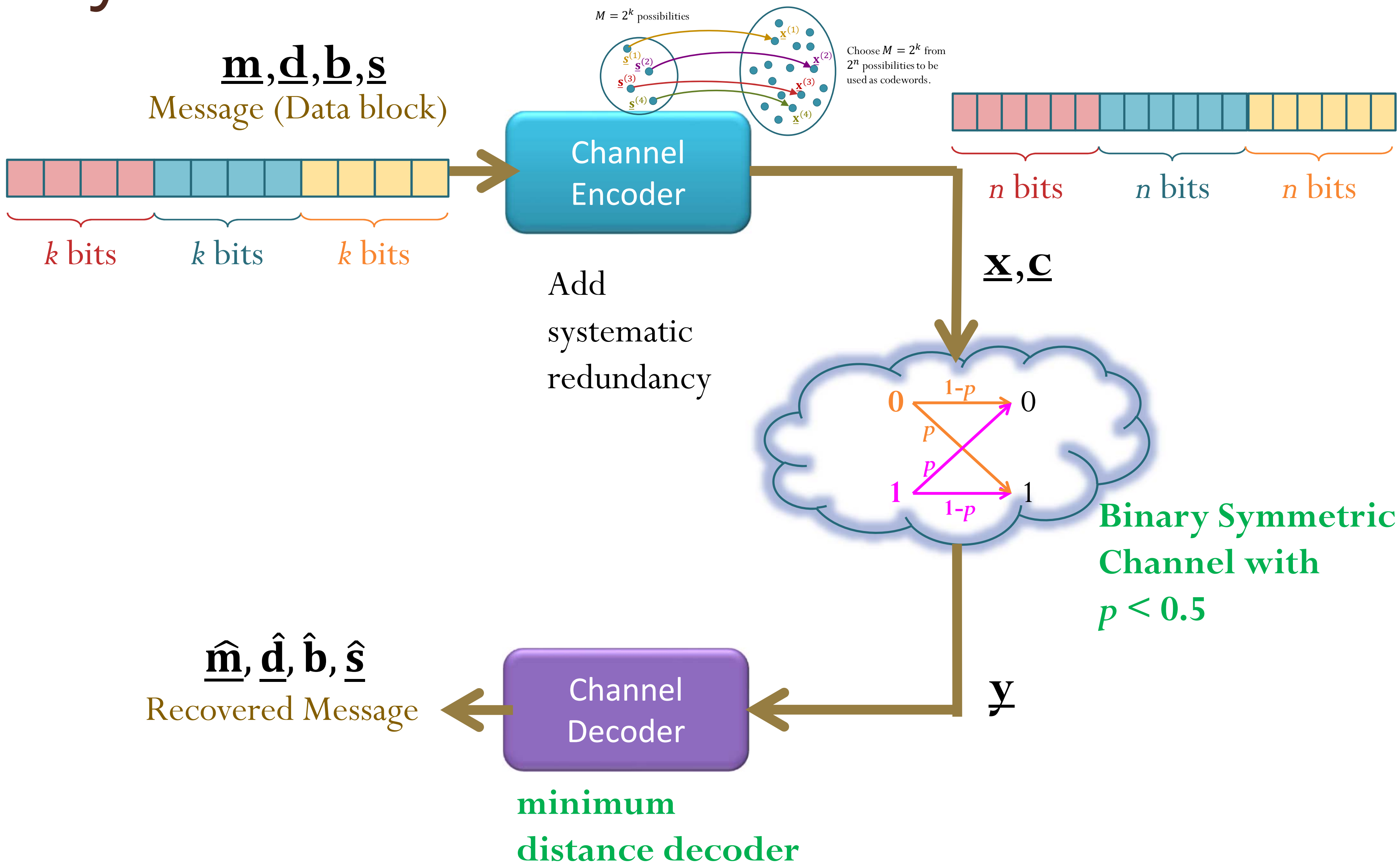


- **(n,k) codes**: n-bit blocks are used to conveys k-info-bit blocks
  - n → Code length
  - k → "Dimension" of the code
  - n-bit blocks → codewords
  - k-info-bit blocks → "messages"
- **Assume $n > k$**
- **Rate**: $R = \frac{k}{n}$.

Recall that the capacity of BSC is $C = 1 - H(p)$.
For $p \in (0,1)$, we also have $C \in (0,1)$.
Achievable rate is < 1.

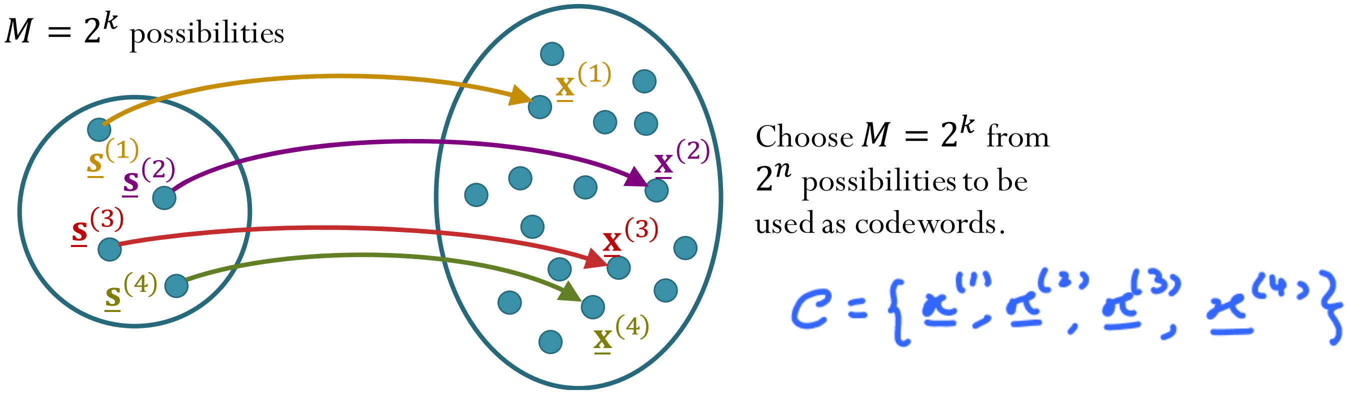
C → Max. achievable rate

System Model for Section 5.1



$\mathcal{C}$

- $\mathcal{C}$ = the collection of all codewords for the code considered.
- Each n-bit block is selected from $\mathcal{C}$.
- The message (data block) has k bits, so there are 2^k possibilities.
- A reasonable code would not assign the same codeword to different messages.
- Therefore, there are 2^k (distinct) codewords in $\mathcal{C}$.



- Ex. Repetition code with $n = 3$

$\mathcal{C} = \{000, 111\}$

MATHEMATICAL SCRIPT CAPITAL C

Charbase

A visual unicode database

Search

← U+1D49D INVALID CHARACTER | U+1D49F MATHEMATICAL SCRIPT CAPITAL D →

U+1D49E: MATHEMATICAL SCRIPT CAPITAL C

G+1 0 Tweet

Your Browser	𝒞
Decomposition	C U+0043
Index	U+1D49E (119966)
Class	Uppercase Letter (Lu)
Block	Mathematical Alphanumeric Symbols
Java Escape	"\ud835\udc9e"
Javascript Escape	"\ud835\udc9e"
Python Escape	u'\U0001d49e'
HTML Escapes	𝒞 𝒞
URL Encoded	q=%F0%9D%92%9E
UTF8	f0 9d 92 9e
UTF16	d835 dc9e

[http://www.charbase.com/1d49e-unicode-mathematical-script-capital-c]

GF(2)

- The construction of the codes can be expressed in matrix form using the following definition of **addition** and **multiplication** of bits:

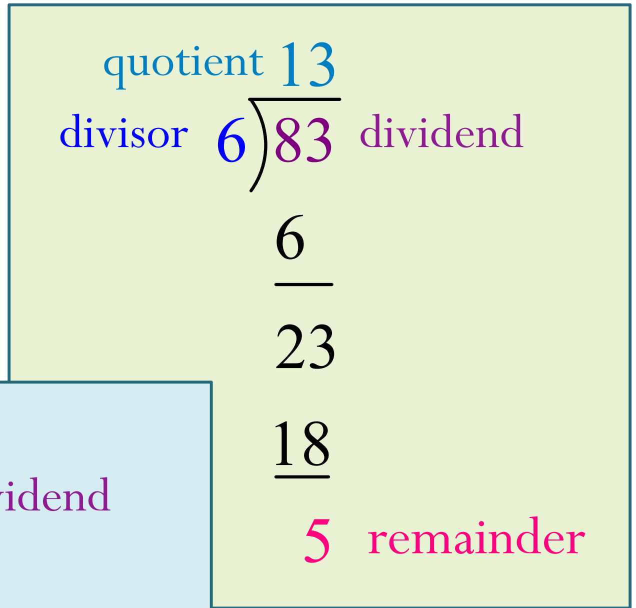
$\oplus$	0	1
0	0	1
1	1	0

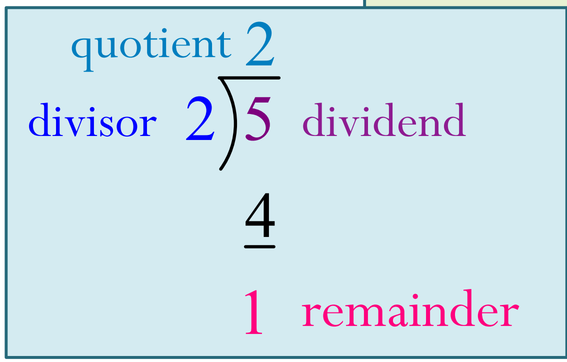
$\cdot$	0	1
0	0	0
1	0	1

- These are **modulo-2** addition and **modulo-2** multiplication, respectively.
- The operations are the same as the **exclusive-or** (**XOR**) operation and the **AND** operation.
  - We will simply call them addition and multiplication so that we can use a matrix formalism to define the code.
- The two-element set {0, 1} together with this definition of addition and multiplication is a number system called a **finite field** or a **Galois field**, and is denoted by the label **GF(2)**.

Modulo operation

- The **modulo operation** finds the **remainder** after division of one number by another (sometimes called **modulus**).
- Given two positive numbers, a (the **dividend**) and n (the **divisor**),
- ***a* modulo *n*** (abbreviated as ***a* mod *n***) is the remainder of the division of a by n.
- "83 mod 6" = 5
- "5 mod 2" = 1
  - In MATLAB, `mod(5,2) = 1`.
- **Congruence relation**
  - $5 \equiv 1 \pmod{2}$





GF(2) and modulo operation

- Normal addition and multiplication (for 0 and 1):

+	0	1
0	0	1
1	1	2

×	0	1
0	0	0
1	0	1

mod 2

- Addition and multiplication in GF(2):

⊕	0	1
0	0	1
1	1	0

•	0	1
0	0	0
1	0	1

GF(2)

- The construction of the codes can be expressed in matrix form using the following definition of addition and multiplication of bits:

$\oplus$	0	1
0	0	1
1	1	0

$\bullet$	0	1
0	0	0
1	0	1

- Note that

$$x \oplus 0 = x$$

$$x \oplus 1 = \overline{x}$$

$$x \oplus x = 0$$

$0 \oplus 0 = 0$
$1 \oplus 0 = 1$
$0 \oplus 1 = 1$
$1 \oplus 1 = 0$
$0 \oplus 0 = 0$
$1 \oplus 1 = 0$

"$\oplus 1$" flips the bit ⇐

check for difference:

$x \oplus y = 1$ iff $x \neq y$

$x \oplus y = 0$ iff $x = y$

The property above implies $-x = x$ ⇒ "subtraction = addition"

By definition, "$-x$" is something that, when added with x, gives 0.

- Extension: For vector and matrix, apply the operations to the elements the same way that addition and multiplication would normally apply (except that the calculations are all in GF(2)).

Examples

- Normal vector addition:

$$
\begin{array}{rl}
 & [1 \quad -1 \quad 2 \quad 1] \; + \\
 & [-2 \quad 3 \quad 0 \quad 1] \\
\hline
= & [-1 \quad 2 \quad 2 \quad 2]
\end{array}
$$

- Vector addition in GF(2):

$$
\begin{array}{rl}
 & [1 \quad 0 \quad 1 \quad 1] \; \oplus \\
 & [0 \quad 1 \quad 0 \quad 1] \\
\hline
= & [1 \quad 1 \quad 1 \quad 0]
\end{array}
$$

Alternatively, one can also apply normal vector addition first, then apply "mod 2" to each element:

$$
\begin{array}{rl}
 & [1 \quad 0 \quad 1 \quad 1] \; + \\
 & [0 \quad 1 \quad 0 \quad 1] \\
\hline
= & [1 \quad 1 \quad 1 \quad 2]
\end{array}
$$

$\downarrow$ mod 2

$$[1 \quad 1 \quad 1 \quad 0]$$

Examples

- Normal matrix multiplication:

$$(7 \times (-2)) + (4 \times 3) + (3 \times (-7)) = -14 + 12 + (-21)$$

$$\begin{bmatrix} 7 & 4 & 3 \\ 2 & 5 & 6 \\ 1 & 8 & 9 \end{bmatrix} \begin{bmatrix} -2 & 4 \\ 3 & -8 \\ -7 & 6 \end{bmatrix} = \begin{bmatrix} -23 & 14 \\ -31 & 4 \\ -41 & -6 \end{bmatrix}$$

- Matrix multiplication in GF(2):

$$(1 \cdot 1) \oplus (0 \cdot 0) \oplus (1 \cdot 1) = 1 \oplus 0 \oplus 1$$

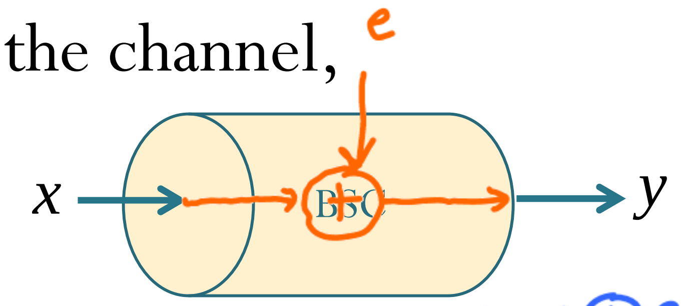
$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}$$

Alternatively, one can also apply normal matrix multiplication first, then apply "mod 2" to each element:

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 0 \\ 2 & 2 \end{bmatrix} \xrightarrow{\text{mod } 2} \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}$$

BSC and the Error Pattern

- For one use of the channel,

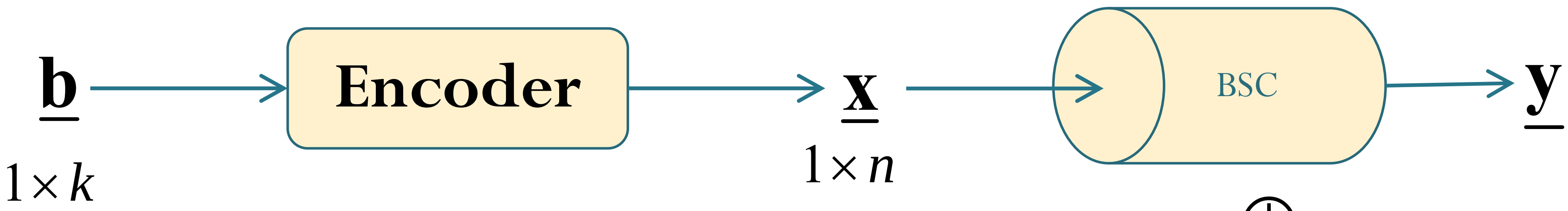


$y = x \oplus e$

Add $e = 0$: output is the same as the input

Add $e = 1$: output is different from the input

- Again, to transmit k information bits, the channel is used n times.



$\underline{\mathbf{y}} = \underline{\mathbf{x}} \oplus \underline{\mathbf{e}}$

error pattern

Its nonzero elements mark the positions of transmission error in $\underline{\mathbf{y}}$

Ex: $\underline{x} = 01010$

$\underline{y} = 01111$

$\underline{e} = 00101$

$\underline{x} \oplus \underline{y} = \underline{x} \oplus \underline{x} \oplus \underline{e} = \underline{e}$

Additional Properties in GF(2)

- The following statements are equivalent
  1. $a \oplus b = c$
  2. $a \oplus c = b$
  3. $b \oplus c = a$

Having one of these is the same as having all three of them.

- The following statements are equivalent
  1. $\underline{\mathbf{a}} \oplus \underline{\mathbf{b}} = \underline{\mathbf{c}}$
  2. $\underline{\mathbf{a}} \oplus \underline{\mathbf{c}} = \underline{\mathbf{b}}$
  3. $\underline{\mathbf{b}} \oplus \underline{\mathbf{c}} = \underline{\mathbf{a}}$

Having one of these is the same as having all three of them.

- In particular, because $\underline{\mathbf{x}} \oplus \underline{\mathbf{e}} = \underline{\boldsymbol{y}}$, if we are given two quantities, we can find the third quantity by summing the other two.

Linear Block Codes

- Definition: $\mathcal{C}$ is a **(binary) linear (block) code** if and only if $\mathcal{C}$ forms a vector (sub)space (over GF(2)).

  In case you forgot about the concept of vector space,…
  - Equivalently, this is the same as requiring that

    $$\text{if } \underline{\mathbf{x}}^{(1)} \text{ and } \underline{\mathbf{x}}^{(2)} \in \mathcal{C}, \text{ then } \underline{\mathbf{x}}^{(1)} \oplus \underline{\mathbf{x}}^{(2)} \in \mathcal{C}.$$

  - Note that any (non-empty) linear code $\mathcal{C}$ must contain $\underline{\mathbf{0}}$.

    Take any $\underline{x} \in \mathcal{C}$. By defn., must have $\underbrace{\underline{x} \oplus \underline{x}}_{\underline{0}} \in \mathcal{C}$

- Ex. The code that we considered in **Problem 5 of HW4** is

$$\mathcal{C} = \{00000, 01000, 10001, 11111\}$$

Is it a linear code?

Prerequisite: None
Special study on current topics related to Information and Communication Technology.

ITS496 Special Studies in Information Technology II 3(3-0-6)
Prerequisite: None
Special study on current topics related to Information and Communication Technology.

ITS497 Special Studies in Information Technology III 2(2-0-4)
Prerequisite: None
Special studies on current topics related to Information and Communication Technology.

ITS499 Extended Information Technology Training 6(0-40-0)
Prerequisite: Senior standing or consent of Head of School
Extensive on-the-job training of at least 16 weeks (640 hours) at a selected organization that provides information technology services. An individual comprehensive research or practical project must be conducted under close supervision of faculty members and supervisors assigned by the training organization. At the end of the training, the student must submit a report of the project and also give a presentation.

MAS116 Mathematics I 3(3-0-6)
Prerequisite: None
Mathematical induction; functions; limits; continuity; differential calculus: derivatives of functions, higher order derivatives, extrema, applications of derivatives, indeterminate forms; integral calculus: integrals of functions, techniques of integration, numerical integration, improper integrals; introduction to differential equations and their applications; sequence and series: Taylor's expansion, infinite sums.

MAS117 Mathematics II 3(3-0-6)
Prerequisite: Have earned credits of MAS116 or consent of Head of School
Analytic geometry in calculus: polar and curvilinear coordinates; three-dimensional space: vectors, lines, planes, and surfaces in three-dimensional space; function of several variables; calculus of real-valued functions of several variables and its applications; partial derivatives, extremes of functions, functions of higher derivatives, Lagrange multipliers; topics in vector calculus: line and surface integrals, Green's theorem.

MAS210 Mathematics III 3(3-0-6)
Prerequisite: Have earned credits of MAS117 or consent of Head of School
Linear algebra: vector spaces, linear transformation, matrices, determinants, systems of linear equations, Gaussian elimination, eigenvalue problems, eigenvalues and eigenvectors, diagonalization, complex matrices; introduction to complex analysis: complex numbers, analytic functions, complex integration, conformal mapping; calculus of variations; introduction to tensor analysis: Cartesian tensors and their algebra.

MAS215 Differential Equations 3(3-0-6)
Prerequisite: Have earned credits of MAS117 or consent of Head of School
Ordinary differential equations of the first order; linear ordinary differential equations of higher order: matrix notation, homogeneous solutions, method of variation of parameters; general ordinary differential equations: series solutions, Bessel functions, Laplace transformation; Fourier analysis: Fourier series, integrals and transforms; partial differential equations: method of separating variables, applications of Laplace and Fourier transforms; applications to initial-value and boundary: value problems.

MES211 Thermofluids 3(3-0-6)
Prerequisite: Have earned credits of (SCS138 or GTS121) or consent of Head of School
Fundamental concepts in thermodynamics. The first and second law of thermodynamics. Basic concepts and basic properties of fluids. Fundamentals of fluid statics. Fundamentals of fluid dynamics. Characteristics of fluids such as laminar and turbulent flows.

MES231 Engineering Mechanics 3(3-0-6)
(For non-mechanical engineering students)
Prerequisite: Have earned credits of SCS138 or consent of Head of School
Force systems; resultants; equilibrium; trusses; frames and machines; internal force diagrams; mass and geometric properties of objects; fluid statics; kinematics and kinetics of particles and rigid bodies; Newton's second law of motion; work and energy, impulse and momentum.

MES300 Engineering Drawing 3(2-3-4)
Prerequisite: None
Introduction to basic principle of engineering drawing, including lettering, applied geometry, orthographic drawing and sketching, sectional views and conventions, detail drawing, assembly drawing, dimensioning, three dimensioning, basic descriptive geometry dealing with points, lines & planes and their relationships in space and basic developed views. Introduction to Computer Graphics.

MES302 Introduction to Computer Aided Design 2(1-3-2)
Prerequisite: Have earned credits of MES300 or consent of Head of School
Use of industrial Computer Aided Design software for detail design and drafting in various engineering fields such as in

MAS210 Mathematics III 3(3-0-6)
Prerequisite: Have earned credits of MAS117 or consent of Head of School
Linear algebra: **vector spaces**, linear transformation, **matrices**, determinants, systems of linear equations, Gaussian elimination, eigenvalue problems, eigenvalues and eigenvectors, diagonalization, complex matrices; introduction to complex analysis: complex numbers, analytic functions, complex integration, conformal mapping; calculus of variations; introduction to tensor analysis: Cartesian tensors and their algebra.

Ex. Checking Linearity

- $\mathcal{C} = \{00000, 01000, 10001, 11111\}$
- Step 1: Check that $\mathbf{0} \in \mathcal{C}$.
  - OK for this example.
- Step 2: Check that

$$\text{if } \underline{\mathbf{x}}^{(1)} \text{ and } \underline{\mathbf{x}}^{(2)} \in \mathcal{C}, \text{ then } \underline{\mathbf{x}}^{(1)} \oplus \underline{\mathbf{x}}^{(2)} \in \mathcal{C}.$$

$\oplus$	00000	01000	10001	11111
00000	00000 ✓	01000 ✓	10001 ✓	11111 ✓
01000	✓	00000	11001 ✗	
10001	✓		00000	
11111	✓			00000

11001 $\notin \mathcal{C}$ $\Rightarrow$ not linear

Ex. Checking Linearity

- $\mathcal{C} = \{00000, 01000, 10001, 11111\}$
- Step 1: Check that $\mathbf{0} \in \mathcal{C}$.
  - OK for this example.
- Step 2: Check that

$$\text{if } \underline{\mathbf{x}}^{(1)} \text{ and } \underline{\mathbf{x}}^{(2)} \in \mathcal{C}, \text{ then } \underline{\mathbf{x}}^{(1)} \oplus \underline{\mathbf{x}}^{(2)} \in \mathcal{C}.$$

  - Here, we have many counter-examples. So, the code is **not linear**.

$\oplus$	00000	01000	10001	11111
00000	00000	01000	10001	11111
01000	01000	00000	11001	10111
10001	10001	11001	00000	01110
11111	11111	10111	01110	00000

Checking Linearity

- Step 1: Check that $\mathbf{0} \in \mathcal{C}$.
- Step 2: Check that

  $$\text{if } \underline{\mathbf{x}}^{(1)} \text{ and } \underline{\mathbf{x}}^{(2)} \in \mathcal{C}, \text{ then } \underline{\mathbf{x}}^{(1)} \oplus \underline{\mathbf{x}}^{(2)} \in \mathcal{C}.$$

  - It may seem that we need to check $|\mathcal{C}|^2$ pairs.
  - Actually, we need to check only $\binom{n-1}{2}$ pairs.

$\oplus$	00000	01000	10001	11111
00000	00000	01000	10001	11111
01000	01000	00000	11001	10111
10001	10001	11001	00000	01110
11111	11111	10111	01110	00000

$\underline{\mathbf{x}} \oplus \underline{\mathbf{0}} = \underline{\mathbf{x}}$

$\underline{\mathbf{x}} \oplus \underline{\mathbf{x}} = \underline{\mathbf{0}}$

$\underline{\mathbf{x}}^{(1)} \oplus \underline{\mathbf{x}}^{(2)} = \underline{\mathbf{x}}^{(2)} \oplus \underline{\mathbf{x}}^{(1)}$

Ex. ~~Checking~~ Creating Linearity

- We have checked that
  $\mathcal{C} = \{00000, \sout{01000}\ 01110, 10001, 11111\}$
  is not linear.
- Change one codeword in $\mathcal{C}$ to make the code linear.

$\oplus$	00000			
00000				

Solution

Ex. Checking Linearity

- We have checked that $\mathcal{C} = \{00000, 01000, 10001, \sout{11111}\}$ (11001) is not linear.
- Change one codeword in $\mathcal{C}$ to make the code linear.

For linearity, we always need **0**

If we want these two to be in our code, then their sum must be in our code too. So, we change 11111 to 11001.

⊕	00000	01000	10001	11001
00000				
01000			11001	10001
10001				01000
11111				

Solution

Ex. Checking Linearity

- We have checked that
  $\mathcal{C} = \{00000, 01000, 10001, 11111\}$
  is not linear.
- Change one codeword in $\mathcal{C}$ to make the code linear.
- Three solutions:
  - $\mathcal{C}$ = {00000, 01000, 10001, ~~11111~~ → 11001}
  - $\mathcal{C}$ = {00000, 01000, ~~10001~~ → 10111, 11111}
  - $\mathcal{C}$ = {00000, ~~01000~~ → 01110, 10001, 11111}

⊕	00000	01000	10001	11111
00000	00000	01000	10001	11111
01000	01000	00000	11001	10111
10001	10001	11001	00000	01110
11111	11111	10111	01110	00000

Linear Block Codes: Motivation (1)

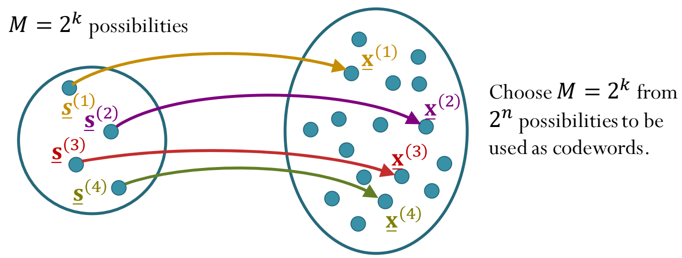
- Why linear block codes are popular?
- Recall: General block **encoding**
  - Characterized by its codebook.
    - The table that lists all the 2^k mapping from the k-bit info-block $\underline{\mathbf{s}}$ to the n-bit codeword $\underline{\mathbf{x}}$ is called the **codebook**.
    - The M info-blocks are denoted by $\underline{\mathbf{s}}^{(1)}, \underline{\mathbf{s}}^{(2)}, \dots, \underline{\mathbf{s}}^{(M)}$. The corresponding M codewords are denoted by $\underline{\mathbf{x}}^{(1)}, \underline{\mathbf{x}}^{(2)}, \dots, \underline{\mathbf{x}}^{(M)}$, respectively.

[See Section 3.5 of the lecture notes.]

k bits n bits

2^k rows

index i	info-block $\underline{\mathbf{s}}$	codeword $\underline{\mathbf{x}}$
1	$\underline{\mathbf{s}}^{(1)} = 000\dots0$	$\underline{\mathbf{x}}^{(1)} =$
2	$\underline{\mathbf{s}}^{(2)} = 000\dots1$	$\underline{\mathbf{x}}^{(2)} =$
$\vdots$	$\vdots$	$\vdots$
M	$\underline{\mathbf{s}}^{(M)} = 111\dots1$	$\underline{\mathbf{x}}^{(M)} =$



  - Can be realized by combinational/combinatorial circuit.
    - If lucky, can used K-map to simplify the circuit.

Linear Block Codes: Motivation (2)

- Why linear block codes are popular?
- Linear block encoding is the same as matrix multiplication.
  - See next slide.
  - The matrix replaces the table for the codebook.
  - The size of the matrix is only $k \times n$ bits.
    - Compare this against the table (codebook) of size $2^k \times (k+n)$ bits for general block encoding.
- Linearity $\Rightarrow$ easier implementation and analysis
- Performance of the class of linear block codes is similar to performance of the general class of block codes.
  - Can limit our study to the subclass of linear block codes without sacrificing system performance.

Example

- $\mathcal{C} = \{00000, 01000, 10001, 11001\}$
- Let

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

$g^{(1)}$ $g^{(2)}$

$\underline{b}G = b_1 g^{(1)} \oplus b_2 g^{(2)}$

$\underline{b} = [b_1 \ b_2]$

- Find $\underline{\mathbf{b}}\mathbf{G}$ when $\underline{\mathbf{b}} = [0\ 0]$.
- Find $\underline{\mathbf{b}}\mathbf{G}$ when $\underline{\mathbf{b}} = [0\ 1]$.
- Find $\underline{\mathbf{b}}\mathbf{G}$ when $\underline{\mathbf{b}} = [1\ 0]$.
- Find $\underline{\mathbf{b}}\mathbf{G}$ when $\underline{\mathbf{b}} = [1\ 1]$.

$\underline{b}G = [0\ 0\ 0\ 0\ 0]$

$[1\ 0\ 0\ 0\ 1]$

$[0\ 1\ 0\ 0\ 0]$

$[1\ 1\ 0\ 0\ 1]$

All possible two-bit vectors

Block Matrices

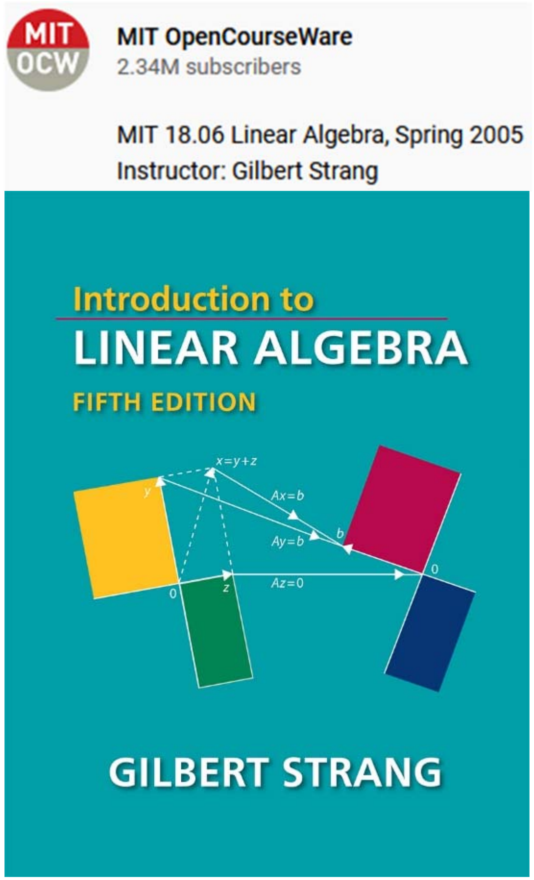
- A **block matrix** or a **partitioned matrix** is a matrix that is interpreted as having been broken into sections called **blocks** or **submatrices**.
- Examples:

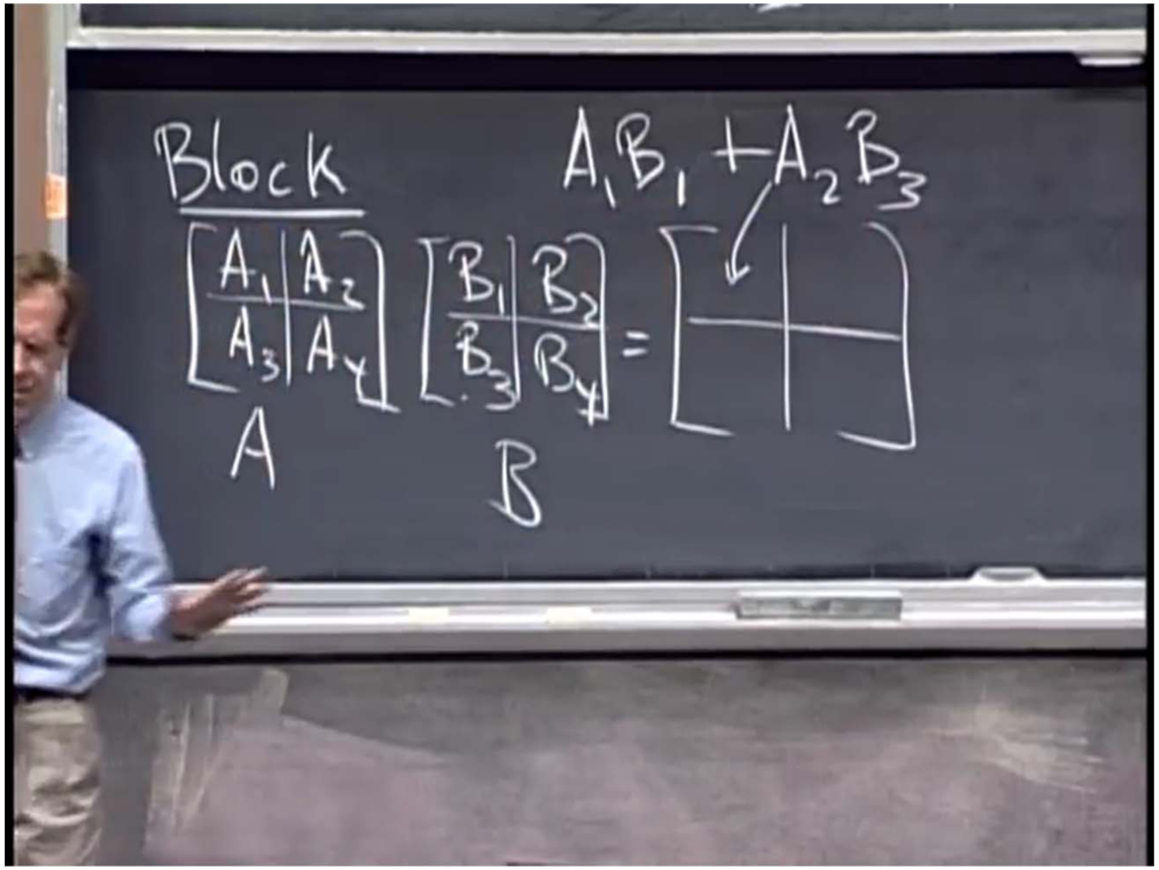
$$\left(\begin{array}{cc|ccc} 10 & 6 & 6 & 4 & 3 \\ 9 & 7 & 3 & 5 & 9 \end{array}\right) = \begin{pmatrix} A & B \end{pmatrix}$$

$$\left(\begin{array}{ccc|ccccc} 2 & 2 & 5 & 10 & 2 & 10 & 2 & 5 \\ 3 & 3 & 4 & 5 & 10 & 5 & 3 & 6 \\ \hline 3 & 3 & 4 & 1 & 1 & 5 & 5 & 6 \\ 7 & 2 & 5 & 3 & 10 & 6 & 10 & 3 \\ 8 & 3 & 6 & 9 & 8 & 3 & 6 & 5 \end{array}\right) = \begin{pmatrix} C & D \\ E & F \end{pmatrix}$$

Block Matrix Multiplications

- Matrix multiplication can also be carried out blockwise (assuming that the block sizes are compatible).





[https://youtu.be/FX4C-JpTFgY?t=1103]

Ex: Block Matrix Multiplications

$$
\left(\underbrace{\begin{matrix}10 & 6\\ 9 & 7\end{matrix}}_{A}\;\underbrace{\begin{matrix}6 & 4 & 3\\ 3 & 5 & 9\end{matrix}}_{B}\right)\times\left(\begin{array}{ccc|ccccc}2 & 2 & 5 & 10 & 2 & 10 & 2 & 5\\ 3 & 3 & 4 & 5 & 10 & 5 & 3 & 6\\ \hline 3 & 3 & 4 & 1 & 1 & 5 & 5 & 6\\ 7 & 2 & 5 & 3 & 10 & 6 & 10 & 3\\ 8 & 3 & 6 & 9 & 8 & 3 & 6 & 5\end{array}\right) \quad \begin{array}{cc} C & D\\ E & F\end{array}
$$

$$
=\left(\underbrace{\begin{matrix}108 & 73 & 136\\ 155 & 85 & 164\end{matrix}}_{AC+BE}\;\underbrace{\begin{matrix}175 & 150 & 193 & 126 & 149\\ 224 & 213 & 197 & 158 & 165\end{matrix}}_{AD+BF}\right)
$$

$$
\underbrace{\begin{pmatrix}10 & 6 & 6 & 4 & 3\\ 9 & 7 & 3 & 5 & 9\end{pmatrix}}_{X}\times\left(\underbrace{\begin{matrix}2 & 2 & 5 & 10\\ 3 & 3 & 4 & 5\\ 3 & 3 & 4 & 1\\ 7 & 2 & 5 & 3\\ 8 & 3 & 6 & 9\end{matrix}}_{G}\;\underbrace{\begin{matrix}2 & 10 & 2 & 5\\ 10 & 5 & 3 & 6\\ 1 & 5 & 5 & 6\\ 10 & 6 & 10 & 3\\ 8 & 3 & 6 & 5\end{matrix}}_{H}\right)
$$

$$
=\left(\underbrace{\begin{matrix}108 & 73 & 136 & 175\\ 155 & 85 & 164 & 224\end{matrix}}_{XG}\;\underbrace{\begin{matrix}150 & 193 & 126 & 149\\ 213 & 197 & 158 & 165\end{matrix}}_{XH}\right)
$$

Linear Block Codes: Generator Matrix

For any linear code, there is a matrix $\mathbf{G} = \begin{bmatrix} \underline{\mathbf{g}}^{(1)} \\ \underline{\mathbf{g}}^{(2)} \\ \vdots \\ \underline{\mathbf{g}}^{(k)} \end{bmatrix}_{k \times n}$

called the **generator matrix**

such that, for any codeword $\underline{\mathbf{x}}$, there is a message vector $\underline{\mathbf{b}}$ which produces $\underline{\mathbf{x}}$ by

$$\underline{\mathbf{x}} = \underline{\mathbf{b}}\mathbf{G} = [b_1 \; b_2 \; \cdots \; b_k] \begin{bmatrix} g^{(1)} \\ g^{(2)} \\ \vdots \\ g^{(k)} \end{bmatrix} = b_1 g^{(1)} \oplus b_2 g^{(2)} \oplus \cdots \oplus b_k g^{(k)}$$

From $\underline{\mathbf{b}}$ to $\underline{\mathbf{x}}$

$$\underline{\mathbf{x}} = \underline{\mathbf{b}}\mathbf{G} = [b_1 \ \ b_2 \ \cdots \ \ b_k] \begin{bmatrix} \underline{\mathbf{g}}^{(1)} \\ \underline{\mathbf{g}}^{(2)} \\ \vdots \\ \underline{\mathbf{g}}^{(k)} \end{bmatrix}_{k \times n}$$

$$= b_1 \underline{\mathbf{g}}^{(1)} \oplus b_2 \underline{\mathbf{g}}^{(2)} \oplus \cdots \oplus b_k \underline{\mathbf{g}}^{(k)} = \sum_{j=1}^{k} b_j \underline{\mathbf{g}}^{(j)}$$

- Any codeword is simply a linear combination of the rows of **G**.
  - The weights are given by the bits in the message $\underline{\mathbf{b}}$

Linear Combination in GF(2)

- A **linear combination** is an expression constructed from a set of terms by multiplying each term by a constant (weight) and adding the results.
- For example, a linear combination of x and y would be any expression of the form $ax + by$, where a and b are constants.
- General expression:

$$c_1 \underline{\mathbf{a}}^{(1)} + c_2 \underline{\mathbf{a}}^{(2)} + \cdots + c_k \underline{\mathbf{a}}^{(k)}$$

- In GF(2), c_i is limited to being 0 or 1. So, a linear combination is simply a sum of a sub-collection of the vectors.

Linear Block Codes: Generator Matrix

For any linear code, there is a matrix $\mathbf{G} = \begin{bmatrix} \underline{\mathbf{g}}^{(1)} \\ \underline{\mathbf{g}}^{(2)} \\ \vdots \\ \underline{\mathbf{g}}^{(k)} \end{bmatrix}_{k \times n}$

called the **generator matrix**

such that, for any codeword $\underline{\mathbf{x}}$, there is a message vector $\underline{\mathbf{b}}$ which produces $\underline{\mathbf{x}}$ by

$$\underline{\mathbf{x}} = \underline{\mathbf{b}}\mathbf{G} = \sum_{j=1}^{k} b_j \underline{\mathbf{g}}^{(j)}$$

mod-2 summation

Note:

(1) Any codeword can be expressed as a linear combination of the rows of $\mathbf{G}$

(2) $C = \{\underline{\mathbf{b}}\mathbf{G} : \underline{\mathbf{b}} \in \{0,1\}^k\}$

Note also that, given a matrix $\mathbf{G}$, the (block) code that is constructed by (2) is always linear.

Fact: If a code is generated by plugging in every possible $\underline{\mathbf{b}}$ into $\underline{\mathbf{x}} = \underline{\mathbf{b}}\mathbf{G}$, then the code will automatically be linear.

Proof

If $\mathbf{G}$ has k rows. Then, $\underline{\mathbf{b}}$ will have k bits. We can list them all as $\underline{\mathbf{b}}^{(1)}, \underline{\mathbf{b}}^{(2)}, \dots, \underline{\mathbf{b}}^{(2^k)}$. The corresponding codewords are

$$\underline{\mathbf{x}}^{(i)} = \underline{\mathbf{b}}^{(i)}\mathbf{G} \text{ for } i = 1, 2, \dots, 2^k.$$

Let's take two codewords, say, $\underline{\mathbf{x}}^{(i_1)}$ and $\underline{\mathbf{x}}^{(i_2)}$. By construction, $\underline{\mathbf{x}}^{(i_1)} = \underline{\mathbf{b}}^{(i_1)}\mathbf{G}$ and $\underline{\mathbf{x}}^{(i_2)} = \underline{\mathbf{b}}^{(i_2)}\mathbf{G}$. Now, consider the sum of these two codewords:

$$\underline{\mathbf{x}}^{(i_1)} \oplus \underline{\mathbf{x}}^{(i_2)} = \underline{\mathbf{b}}^{(i_1)}\mathbf{G} \oplus \underline{\mathbf{b}}^{(i_2)}\mathbf{G} = \left(\underline{\mathbf{b}}^{(i_1)} \oplus \underline{\mathbf{b}}^{(i_2)}\right)\mathbf{G}$$

Note that because we plug in ***every possible*** $\underline{\mathbf{b}}$ to create this code, we know that $\underline{\mathbf{b}}^{(i_1)} \oplus \underline{\mathbf{b}}^{(i_2)}$ should be one of these $\underline{\mathbf{b}}$. Let's suppose $\underline{\mathbf{b}}^{(i_1)} \oplus \underline{\mathbf{b}}^{(i_2)} = \underline{\mathbf{b}}^{(i_3)}$ for some $\underline{\mathbf{b}}^{(i_3)}$. This means

$$\underline{\mathbf{x}}^{(i_1)} \oplus \underline{\mathbf{x}}^{(i_2)} = \underline{\mathbf{b}}^{(i_3)}\mathbf{G}.$$

But, again, by construction, $\underline{\mathbf{b}}^{(i_3)}\mathbf{G}$ gives a codeword $\underline{\mathbf{x}}^{(i_3)}$ in this code. Because the sum of any two codewords is still a codeword, we conclude that the code is linear.

Linear Block Code: Example

$\underline{b}G$

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

- Find the codeword for the message $\underline{\mathbf{b}} = [1\ 0\ 0]$

1 0 0 1 0 1

- Find the codeword for the message $\underline{\mathbf{b}} = [0\ 1\ 1]$

0 1 1 1 0 1

- How many codewords do this code have? $2^3 = 8$

Linear Block Code: Codebook

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

$$\underline{\mathbf{x}} = \underline{\mathbf{b}}\mathbf{G} = (b_1\ b_2\ b_3)\begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$
$$= (b_1, b_2, b_3, b_1 \oplus b_3, b_2 \oplus b_3, b_1 \oplus b_2)$$

$\underline{\mathbf{b}}$			$\underline{\mathbf{x}}$					
0	0	0	0	0	0	0	0	0
0	0	1	0	0	1	1	1	0
0	1	0	0	1	0	0	1	1
0	1	1	0	1	1	1	0	1
1	0	0	1	0	0	1	0	1
1	0	1	1	0	1	0	1	1
1	1	0	1	1	0	1	1	0
1	1	1	1	1	1	0	0	0