938II - Electronics and communication technologies (2023/24)

Basics of digital communication systems

Marco Luise, Giacomo Bacci
{marco.luise, giacomo.bacci}@unipi.it

Basics of digital communication systems

Outline of the lecture

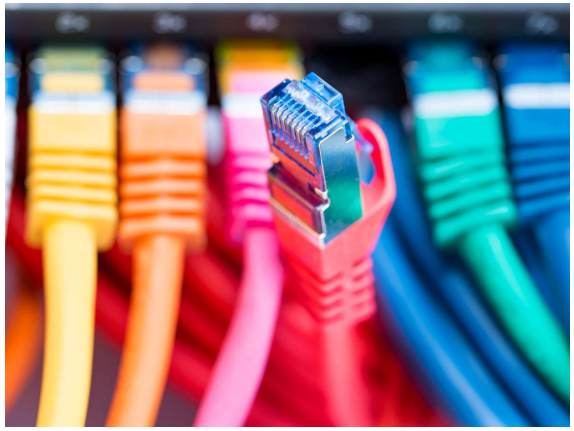
- *Overview of the architecture*
- *Building blocks of transmitter and receiver*
- *Source coding*
- *Channel coding*
- *Modulation techniques*
- *Shannon capacity*
- *Multiplexing and multiple access*

System where there's some source generating info (transmitter) and one that receives it (receiver, sink)

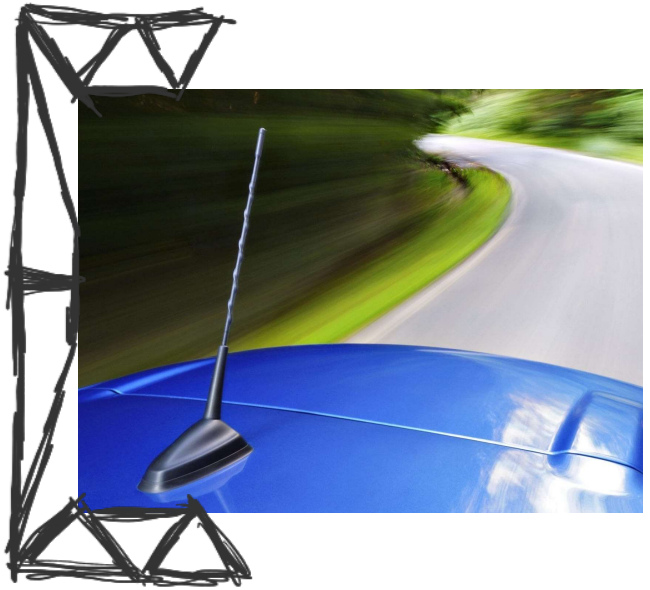
Categories of communication systems (1/3)

First difference in communication systems

- **Wired vs. wireless**



vs.



Typically communication is mixed

- **Broadband vs. narrowband**

Uses a large amount of band.

vs.

NARROWBAND: uses small amount of band, ex. exchanging info with sensors once a week. You don't transfer a lot of information.

The more physical bandwidth the more info you can place on this band.

Categories of communication systems (2/3)

Way in which info is transferred

- **Broadcast (point-to-multipoint) vs. unicast (point-to-point)**

One single point to many subscribers.

vs.

- **Terrestrial vs. satellite** (Terrestrial systems: ex smartphone because they rely purely terrestrial information.

You have very different design requirements.

vs.

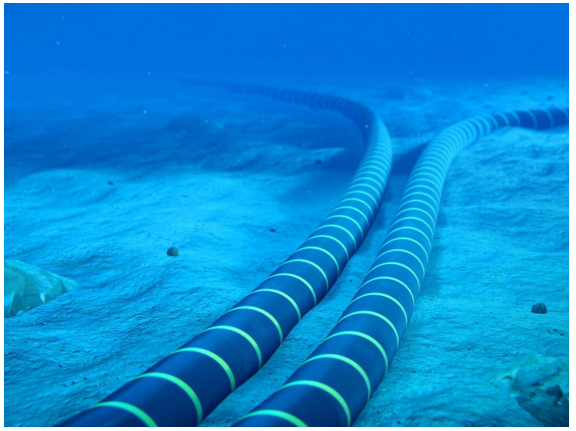
Categories of communication systems (3/3)

- **Access vs. transport**

Some bandwidth requirements

Access networks: the last mile, last part of communication to reach the receiver or transmitter.

vs.



Transport network

… and many more

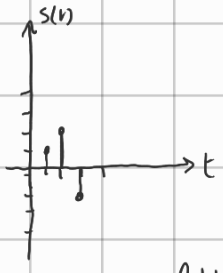
Analog signal: signal which is continuous in time and can assume all the real values.
Continuous in amplitude and time.

Digital signal: we discretize both the time and the amplitude.

We go from Analog to Digital with sampling and quantization

ADVANTAGES OF DIGITAL SIG:

1. REGENERATION: We assign the possible values a quantized signal can assume with bit strings.



$x[0] = 010$. Let's say I need to transmit 010: it's going to be something like:

At the beginning of the fiber. At the receiver, because of the distance we get something like:

Attenuation: the physical mean absorbs some energy

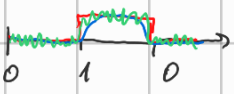
This is the ATTENUATION EFFECT.

Then you have the distorsion effect introduced by a physical medium:

Distortion: mean action is presence of obstacles

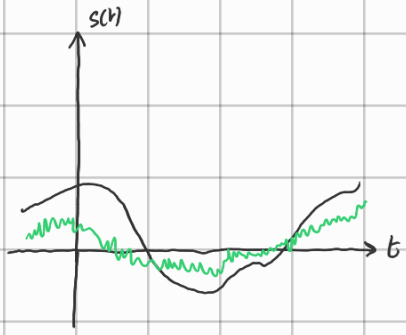
This is the DISTORTION EFFECT.

Then of course we have noise.



Can you still recognise the message? We could put amplifiers during transmitters to decode and regenerate a clean signal with amplifiers in the middle. But if the distance is too much you get something that can't be 100% decoded correctly.

What if I just considered sending an analog signal without digital processing?
We would still have those effects and it would be even worse:



You can't regenerate the original signal!
With a digital signal you can regenerate it!
MAIN ADVANTAGE [You could, with regeneration remove attenuation, but not noise. With equalization you could remove distortion]

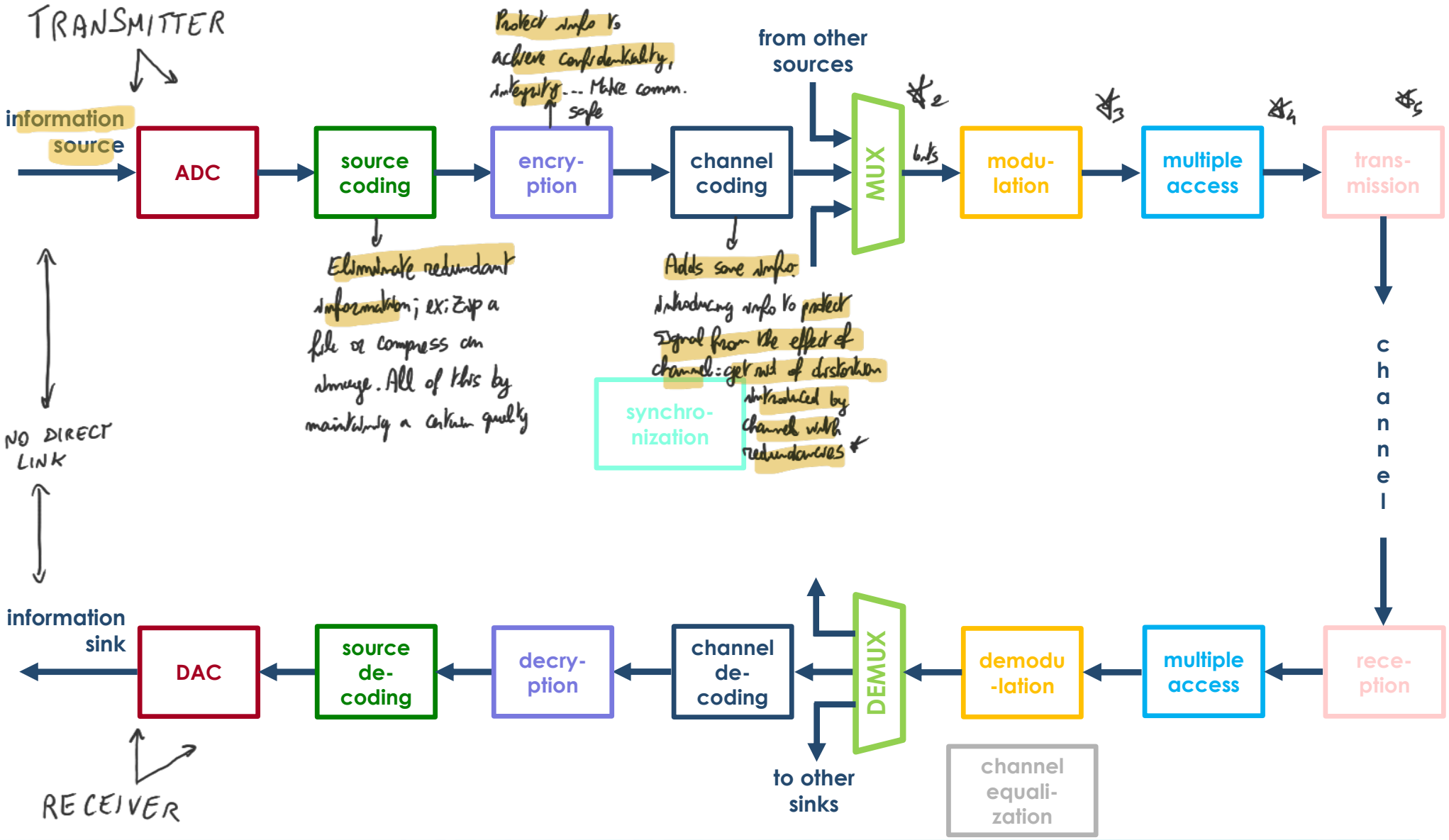
OTHER ADVANTAGES OF USING DIGITAL SIGNALS

- We can treat all different signals in the same way!
- If I'm using digital signals the hardware is cheaper, more efficient and less consuming.
  For example, with analog signals you need linear amplifiers, because you don't want to change the signal. Under certain conditions you can use non linear amplifiers on digital.
- Some operations become unfeasible with analog (encryption)
- It wouldn't make sense for machine to machine communication (digital!) to use analog communication.

WHAT ARE THE DRAWBACKS?

1) In digital system you can have "non graceful degradation", in which you have sudden drop downs in signal qualities
2) I need receiver and sender to be synchronised in order to receive information.
   This happens in analog too, but requires less accuracy than for digital signals.

Elements of a digital communication system



Some of those parts are compulsory (MODULATION). Others no, like ADC, DAC if you already have a digital signal. We could switch position of some blocks too. In general we comply with a certain standard.

* Example of channel coding: Simplest way is send a bit multiple times:

0 → 000 1 → 111 Until I have less than half flips I can reconstruct.

EX: Imagine you want to send 16 bits: 0010110101101101. We rearrange them in a matrix:

We send parity bits for rows and columns:

0	~~0~~ 1	1	0	1
1	1	0	1	1
0	1	1	0	0
1	1	0	1	1
0	1	0	0	

ERROR CORRECTION

If green bit is flipped we identify rows and columns that do not fulfil the parity and fix it. Also, not all the channels behave in the same way. Channel coding has to apply to the specific channel we use.

*2 Aggregation of different strings at the same time with multiplexing techniques in a smart way.

*3 Task is sending bits on your analog channel: transform sequence of bits into a symbol and you have to move them to a certain frequency. For now we are talking about baseband signals. (Signal that has a $f_c = 0$). You cannot transmit a signal like this. The size of an antenna should match the wave wavelength: $\lambda_0 = \frac{c}{f_0} \to +\infty$

Modulation transforms bits into analog signals that can be sent into the channel.

*5: Hardware created to take analog signal and transform it into radio waves (high power amplifiers, antenna, mixers).

*4: Multiple access lets different transmitters coexist in the same system. Allows us to distinguish each users. Different transmitters (different streams for multiplexer) to coexist in the same system.

Remember the relationship: $f_0 = \frac{c}{\lambda_0}$ size of antenna should be the same size of the wave's wavelength.

This is why modulation is needed. It moves digital signal back into the analog domain and to a carrier frequency.

Moving to higher frequencies allows you to use multiple access.

If I properly pair multiple access with modulation, I can have multiple people communicate on the same channel.

1. CHANNEL EQUALIZATION: appears in the receiver chain. The objective is to remove the distortion introduced by the channel. To do so, you need to estimate the channel (you have the channel estimation block that understands the distortion).
2. SYNCHRONIZATION: can be done in different ways, but you need to know when each bit starts.

LEZIONE 5 END

Source coding: Remove the redundancy

Source coding

Source coding aims at data compression, involving encoding information to use fewer bits than the original representation: Why? For storage and cheaper transmission. Maintain the same quality while reducing the file size.

- Lossless compression identifies and eliminates statistical redundancy

- Lossy compression identifies and removes unnecessary information

DIFFERENCE:

1) Start from an $X[n] \rightarrow$ [SOURCE ENCODER] $\rightarrow y[n] \rightarrow$ [SOURCE DECODER] $\rightarrow X[n]$

If I perform then source decoding, I will get exactly $X[n]$ back.

2) Start from an $X[n] \rightarrow$ [SOURCE ENCODER] $\rightarrow y[n] \rightarrow$ [SOURCE DECODER] $\rightarrow \hat{X}[n] \cong X[n]$

You have some loss in the process.

- Lossy is more efficient in the compression than lossless compression. But is irreversible.

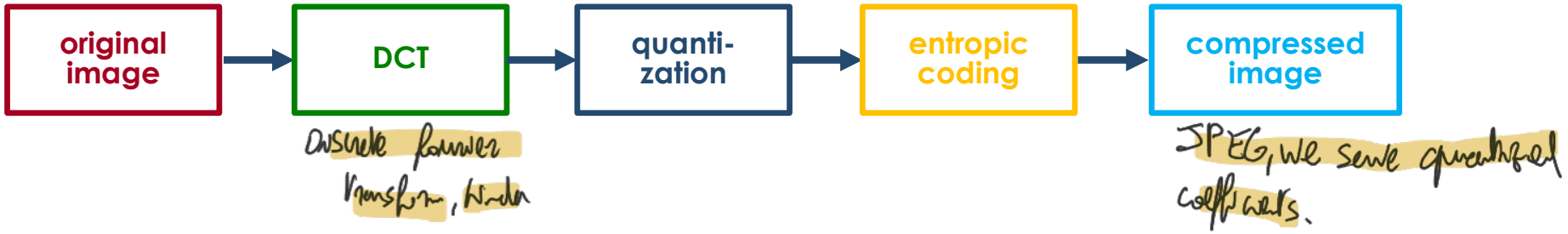
Keep in mind that ADC introduces some loss. So it can make sense to "keep losing something". For native digital signals we use lossless. For analog we use lossy.

An example of lossy compression: image coding (1/7)

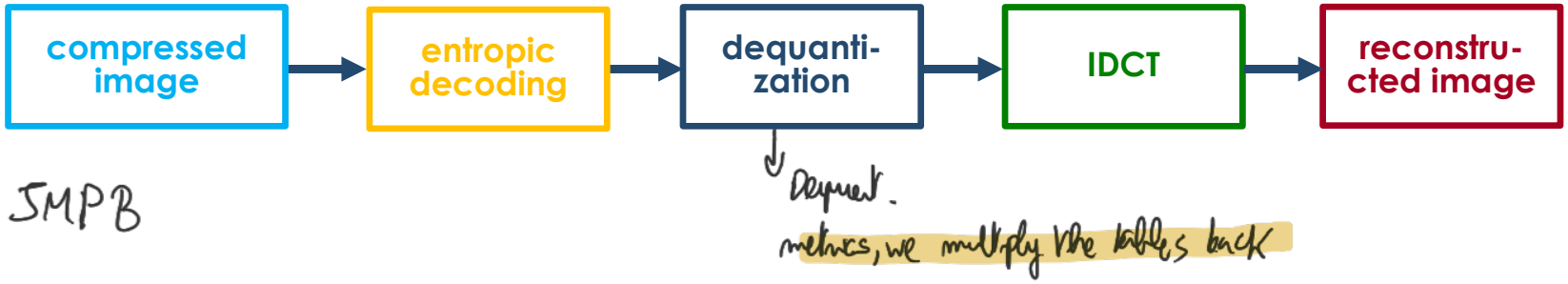
Image coding adopts a lossy compression, which is based on the specific properties of the human vision ⇒ We explain how our brain perceives images.

A

Joint picture expert group (JPEG) encoding scheme:



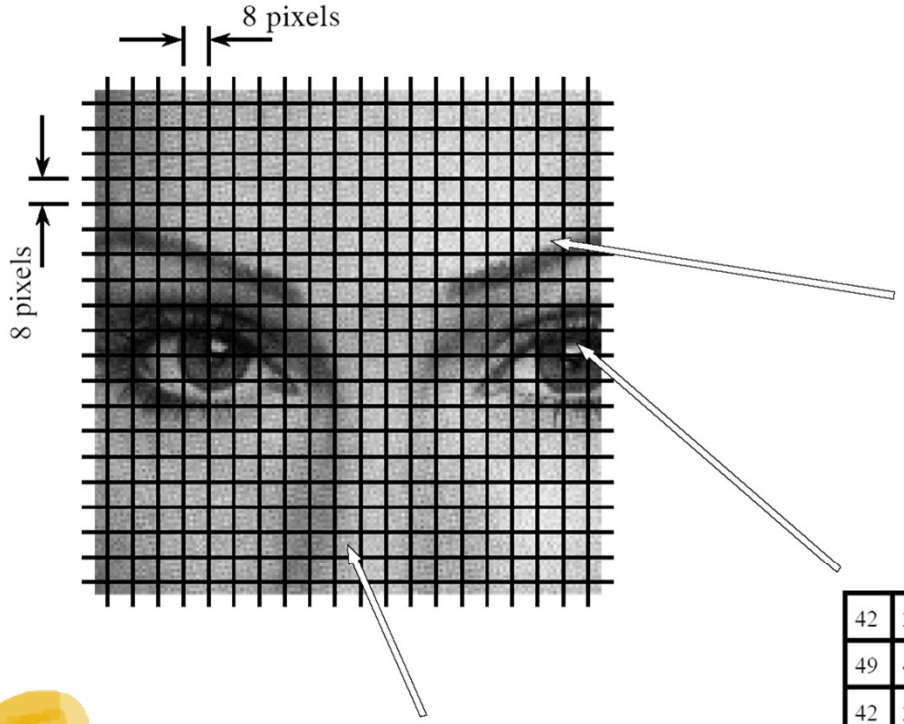
JPEG decoding scheme:



An example of lossy compression: image coding (2/7)

Image segmentation:

Image: bidimensional signal: $X[m_1][m_2]$, value of the grey scale of a point. Each of these points are called pixels: picture elements.

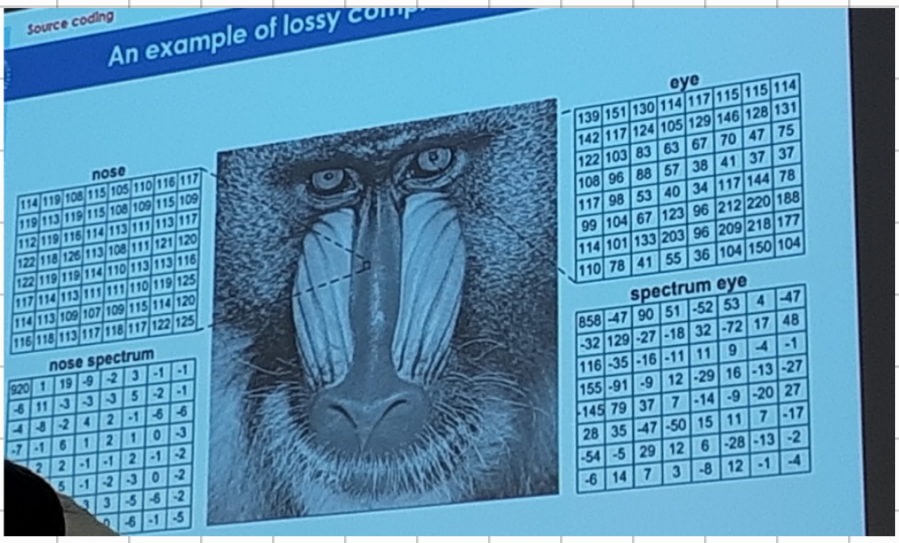


231	224	224	217	217	203	189	196
210	217	203	189	203	224	217	224
196	217	210	224	203	203	196	189
210	203	196	203	182	203	182	189
203	224	203	217	196	175	154	140
182	189	168	161	154	126	119	112
175	154	126	105	140	105	119	84
154	98	105	98	105	63	112	84

42	28	35	28	42	49	35	42
49	49	35	28	35	35	35	42
42	21	21	28	42	35	42	28
21	35	35	42	42	28	28	14
56	70	77	84	91	28	28	21
70	126	133	147	161	91	35	14
126	203	189	182	175	175	35	21
49	189	245	210	182	84	21	35

154	154	175	182	189	168	217	175
154	147	168	154	168	168	196	175
175	154	203	175	189	182	196	182
175	168	168	168	140	175	168	203
133	168	154	196	175	189	203	154
168	161	161	168	154	154	189	189
147	161	175	182	189	175	217	175
175	175	203	175	189	175	175	182

How fast a signal varies in vertical and horizontal axis can be analysed with a bidimensional fourier transform.



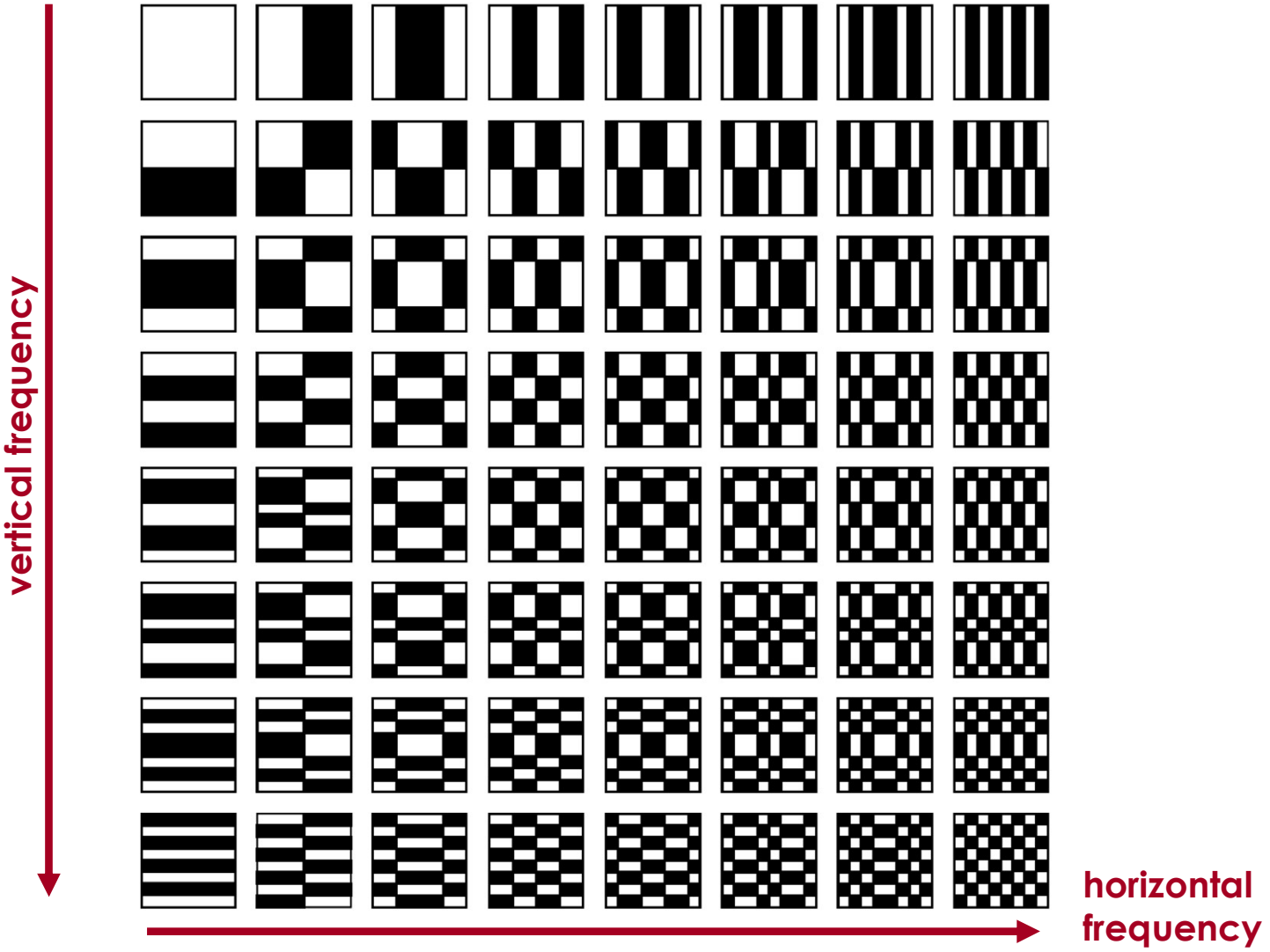
Frequencies in the X axis vs the y axis.

We have very different coefficients in terms of frequencies.

Now I want to distinguish the details of the figure from the rest. Since our vision does not pay attention to details, we can cut the details. We can do this: represent the frequency values in the areas with the details with less bits. So we remove the details without affecting how the mind perceives the image.

An example of lossy compression: image coding (3/7)

Contribution of DCT coefficients to the appearance of the 8×8 pixel block:



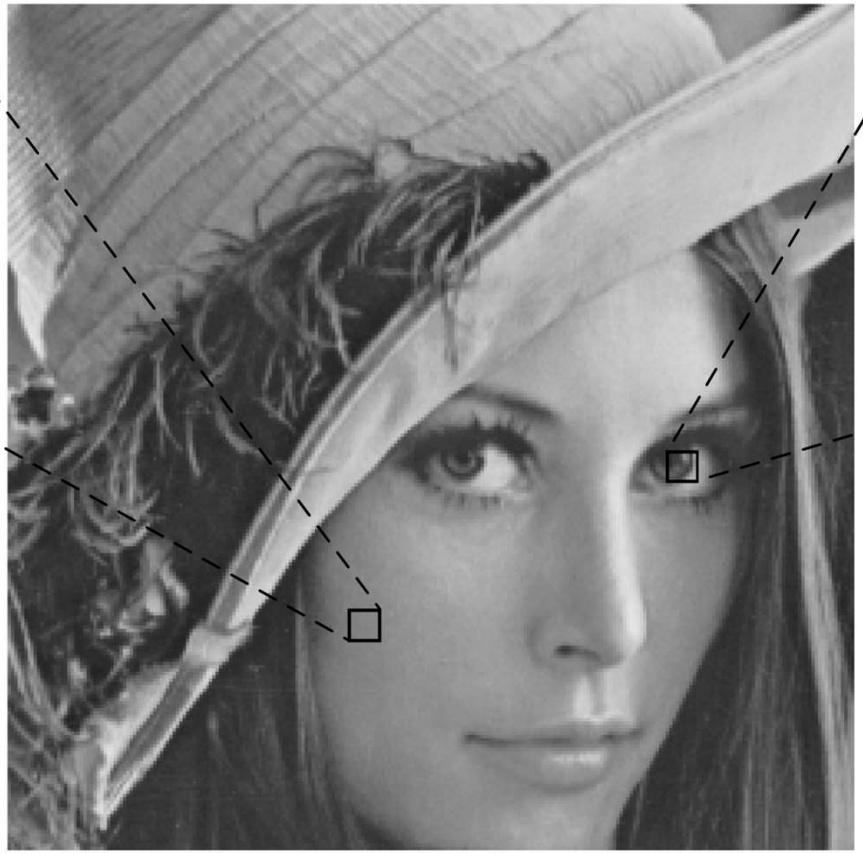
An example of lossy compression: image coding (4/7)

cheek

129	130	131	136	133	133	137	134
130	130	130	136	134	135	136	136
129	128	131	132	132	135	134	136
128	129	130	133	133	134	134	137
130	128	128	132	130	132	134	135
132	128	128	132	131	132	132	136
126	128	130	133	132	132	136	133
126	129	134	133	131	132	134	135

cheek spectrum

1056	−18	−2	−3	3	4	−1	−2
5	−1	−1	2	1	0	−1	−1
2	1	−4	−2	−3	2	−2	1
−1	2	2	2	2	2	−2	−1
0	0	0	−2	−1	0	−1	0
−1	−1	−2	0	−1	0	−1	1
−1	1	2	0	1	−1	3	0
−2	0	0	0	−2	1	0	1



eye

51	60	61	90	144	124	73	157
50	63	64	78	139	133	79	146
47	63	65	70	97	104	73	138
54	74	85	74	84	80	72	145
73	88	103	87	78	64	88	158
87	89	90	88	74	69	119	169
96	97	83	72	74	108	155	174
88	85	77	79	112	152	176	178

eye spectrum

771	−182	60	−46	42	−64	26	−5
−63	3	−84	16	41	−58	28	−9
60	−60	−14	64	−5	−6	−2	−2
5	24	12	−31	21	6	−8	4
8	−1	−13	−6	6	−6	5	3
−15	5	14	−5	−3	4	−2	−4
−7	−3	−3	−3	8	4	−3	−2
−2	5	6	−4	−4	2	0	−2

An example of lossy compression: image coding (5/7)

Quantization tables:

Remove the least sig. bit
least 2 sig. bits

1	1	1	1	1	2	2	4
1	1	1	1	1	2	2	4
1	1	1	1	2	2	2	4
1	1	1	1	2	2	4	8
1	1	2	2	2	2	4	8
2	2	2	2	2	4	8	8
2	2	2	4	4	8	8	16
4	4	4	4	8	8	16	16

1	2	4	8	16	32	64	128
2	4	4	8	16	32	64	128
4	4	8	16	32	64	128	128
8	8	16	32	64	128	128	256
16	16	32	64	128	128	256	256
32	32	64	128	128	256	256	256
64	64	128	128	256	256	256	256
128	128	128	256	256	256	256	256

① ***low compression*** ***high compression***

We apply this matrix here to all the blocks (8×8 pixel). For each box, we do fourier transform and apply the quantization tables.

① Maintain most of the bits, low compression but less loss.
This is a way to cut the details.

Save 0	Save 1	Save 2	Save 3	...			
1	2	4	8	16	32	64	128

Let's say I have: 858. $\Rightarrow \left\lfloor \frac{858}{1} \right\rfloor = 858$ We keep 8 bits.

I have: 90. $\Rightarrow \left\lfloor \frac{90}{4} \right\rfloor = 22$

So you remove some bits. To reconstruct the image, you can remultiply the values by the same quantization tables to get something similar to the original spectrum to with a small quantization error.

JMP A

B

The first coefficient is the only one that needs 11 bits to be saved (in JPG sometimes it is divided by 16).

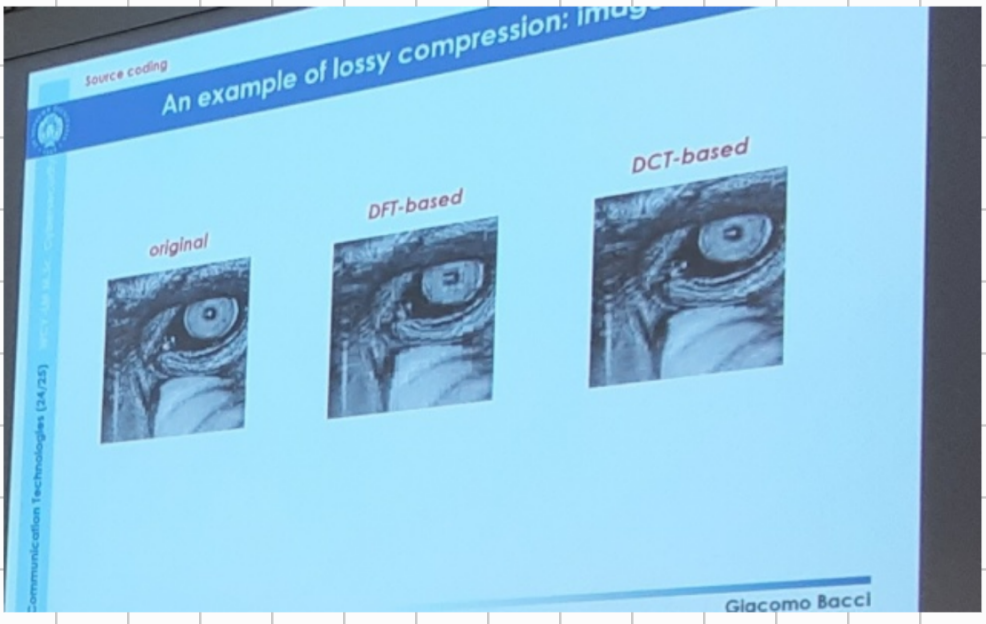
So, for the pixels, the number of bits used in each block is $8 \times 8 \times 8$.
With low compression, you need $8 \times 8 \times 4$ bits.

The other approach is raw compression in the space domain: instead of using X bits for quantization, you use X-y. So less levels to be represented. They are similar.
So what's the advantage of frequency? With high compression you save 75% of bits. With raw quantization it gets lost in the space domains. So 4 values instead of 64 levels. Those two can be represented in the same number of bits. With working in the frequency domain.

Entropic coding: additional boost of (lossless) compression. So in the frequency domain we use even less bits than raw compression. So in JMP A, the only block that introduces loss is QUANTIZATION

Based on the fact that there can be a lot of zeroes so we say "the next one all zeroes". Depending on how we read the file we can optimise how we can compress it losslessly.

JMP A: We use DCT: discrete cosine transform: it is able to better capture the details. The result grain is softer.



An example of lossy compression: image coding (6/7)

50% compression

DCT-based quantization One with the ruffles. *raw quantization*

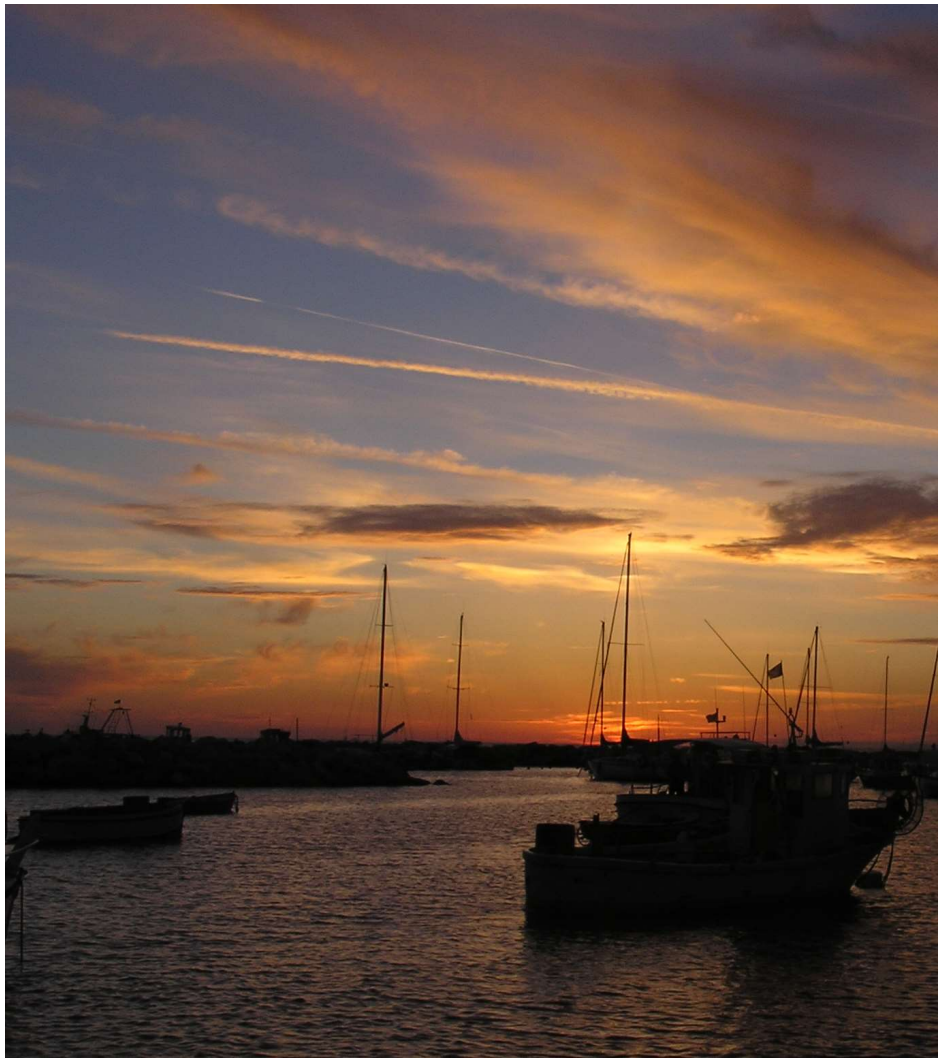
An example of lossy compression: image coding (7/7)

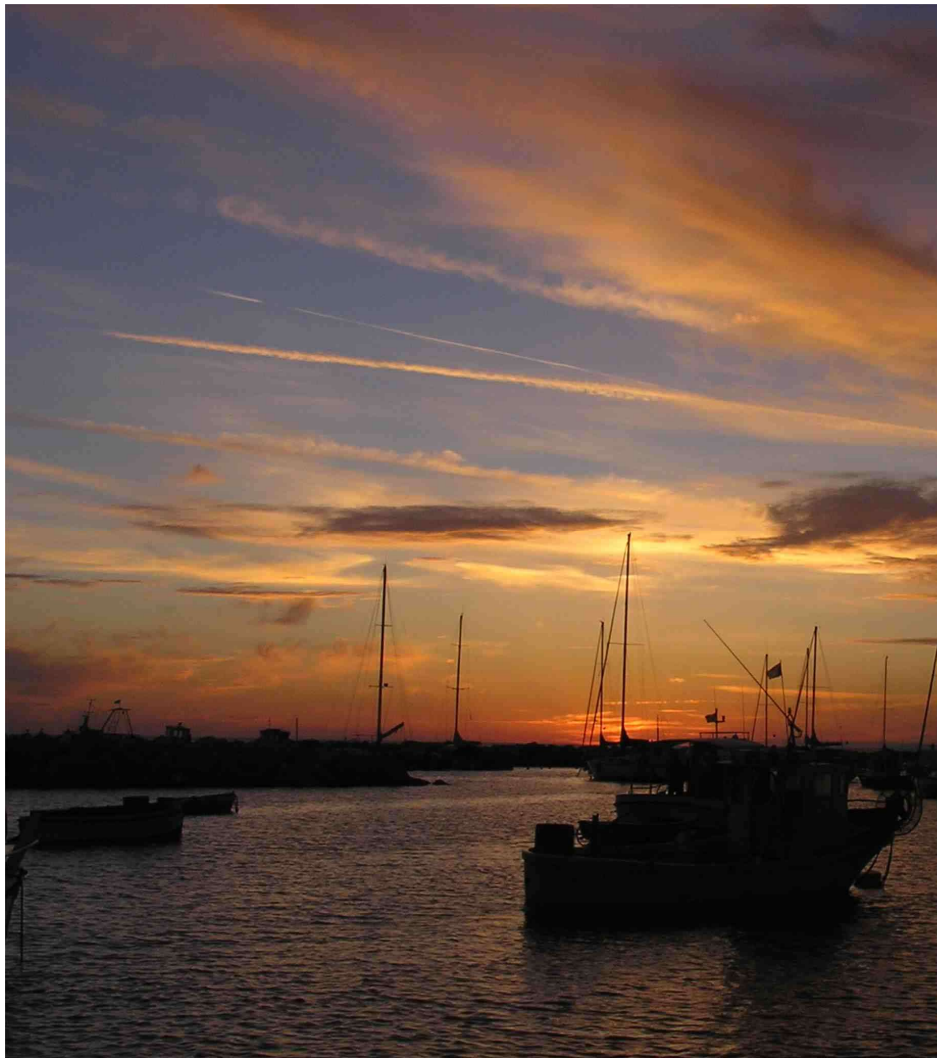
75% compression

DCT-based quantization

raw quantization

Examples of compressed images (1/4)

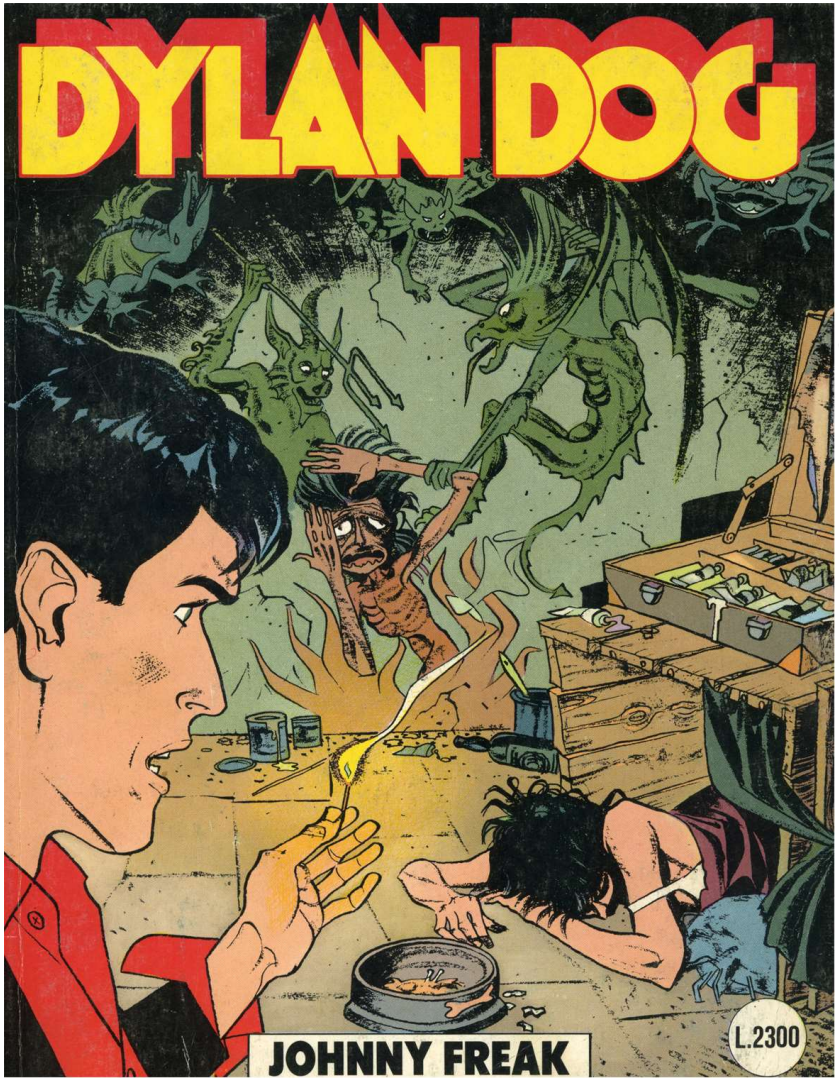




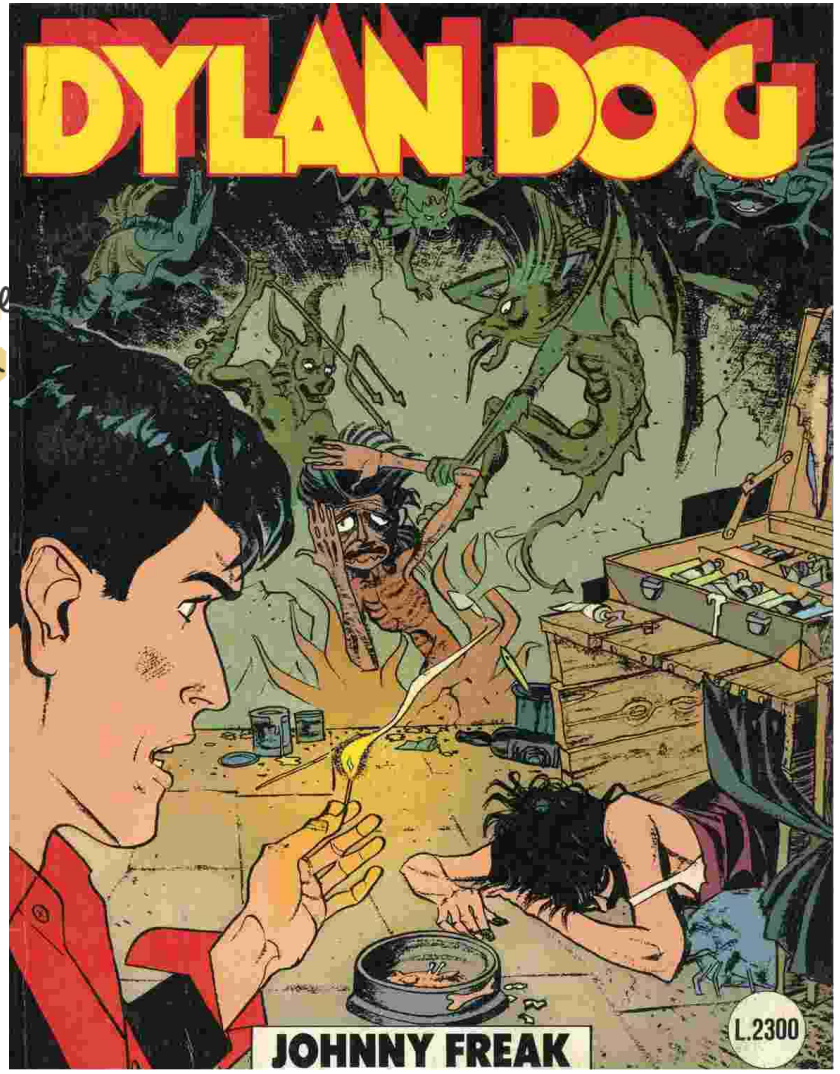
original image 10MB vs 1MB *compressed image (ratio 1:10)*

JPEG WORKS WELL WITH NATURAL IMAGES

Examples of compressed images (2/4)



WHAT IF YOU USE ARTIFICIAL IMAGES? We have more sudden transit. and some artifacts appear.

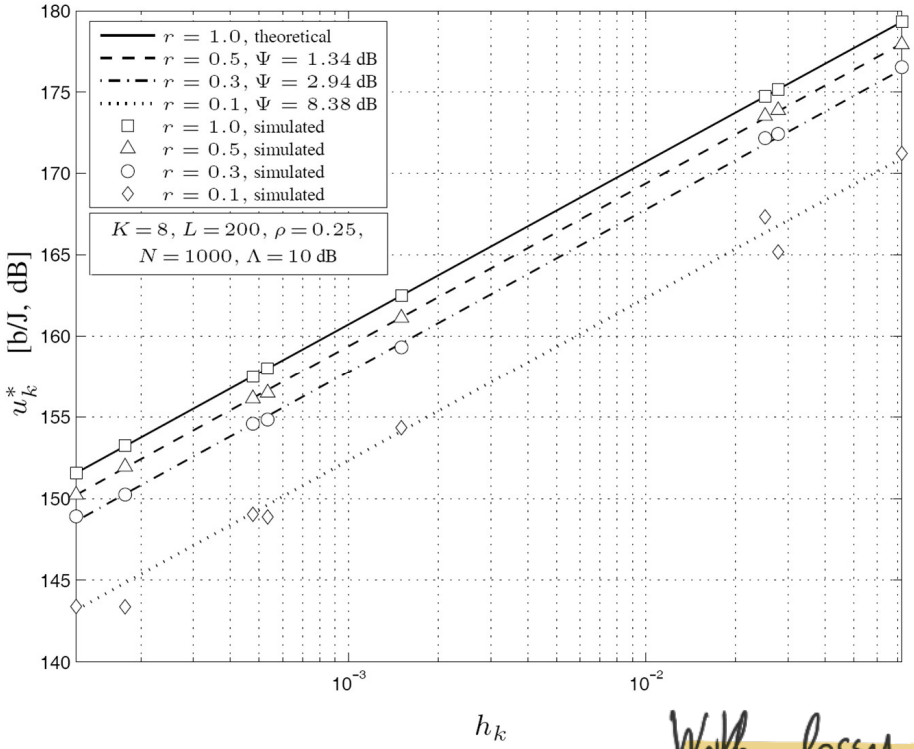


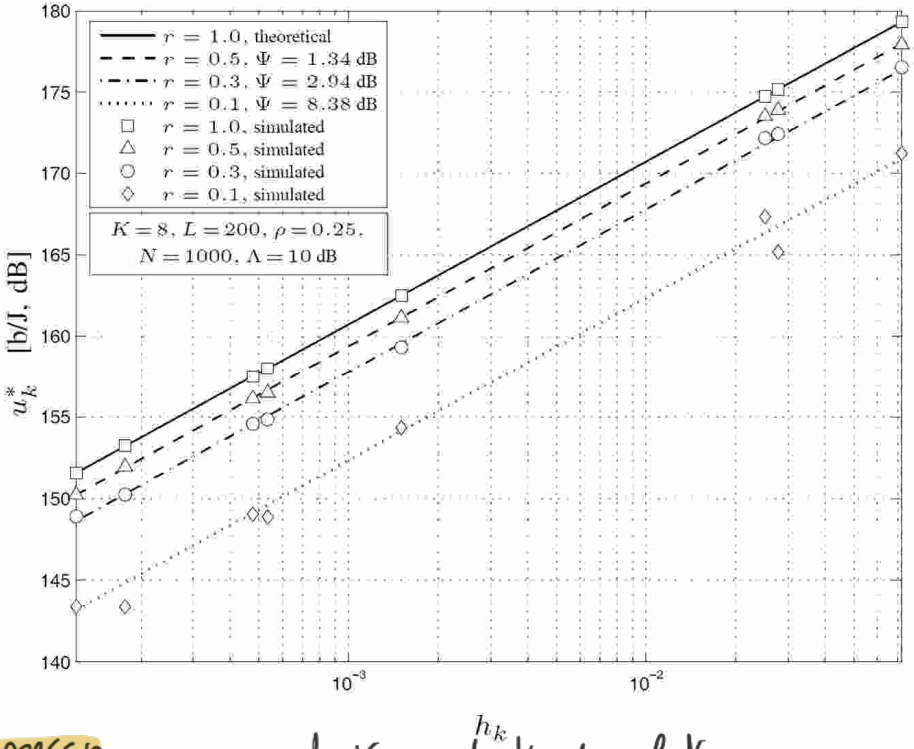
original image

compressed image (ratio 1:10)

Examples of compressed images (3/4)

With artificial images it is worse.





With lossy compression, you need to understand what your output is.

original image *compressed image (ratio 1:10)*

Examples of compressed images (4/4)

8x8 is compromise between efficiency or complexity. Nowadays we are also trying to do it depending on the image type (8x16 or 16x16)

2.66 MB

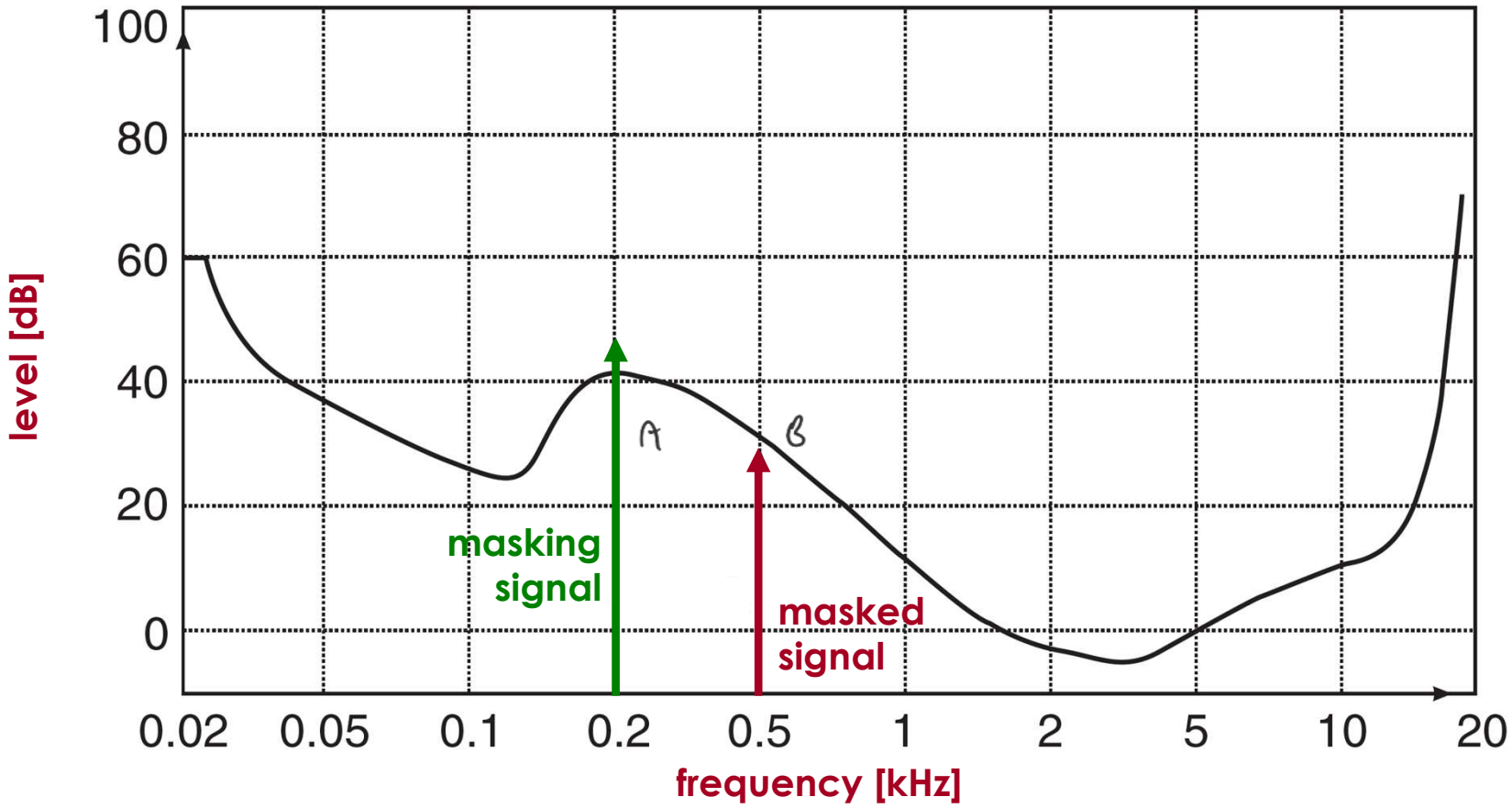
7.02 MB

Basics of audio coding (1/2)

We use lossy compression and investigate how the human ear work.

Similarly, audio coding adopts a lossy compression, again based on the

psycho-acoustic model



Threshold of audible signals. If a certain signal has an amplitude above the threshold, we can hear them, otherwise we don't.

*1 If we send another simusoid before, the ear is changing the threshold so we become deaf to certain frequencies. B was below the detection threshold but when A is added, the threshold changes and we can't hear it anymore. After A stops, we get the threshold back to normal after some time.

The mp3 postprocesses the signal and looks for situations like these and removes frequencies that are still not detectible by the human ear.

Basics of audio coding (2/2)

Audio coding, like image coding, is based on the **semantics** of the source:

- Uncompressed, CD-quality audio (1.41 Mb/s):
- Compressed, MPEG-1/2 audio layer III (MP3) audio (32 kb/s):
- Compressed using JPEG algorithm (32 kb/s):

Why do we need compressed videos?

high-definition video (HD):

- component resolution (R, G, B): 8 bits / component
- color image composition: 3 components / pixel
- resolution: 1920 × 1080 pixels / frame
- image refresh rate: 60 frames / s

$$R_b = (8 \cdot 3 \cdot 1920 \cdot 1080 \cdot 60)\,\mathrm{b/s} \approx 2.99\,\mathrm{Gb/s}$$

ultra-high-definition video (UHD, aka 4k):

- 3840 × 2160 pixels / frame, using 120 f/s refresh rate and 48 bits per pixel

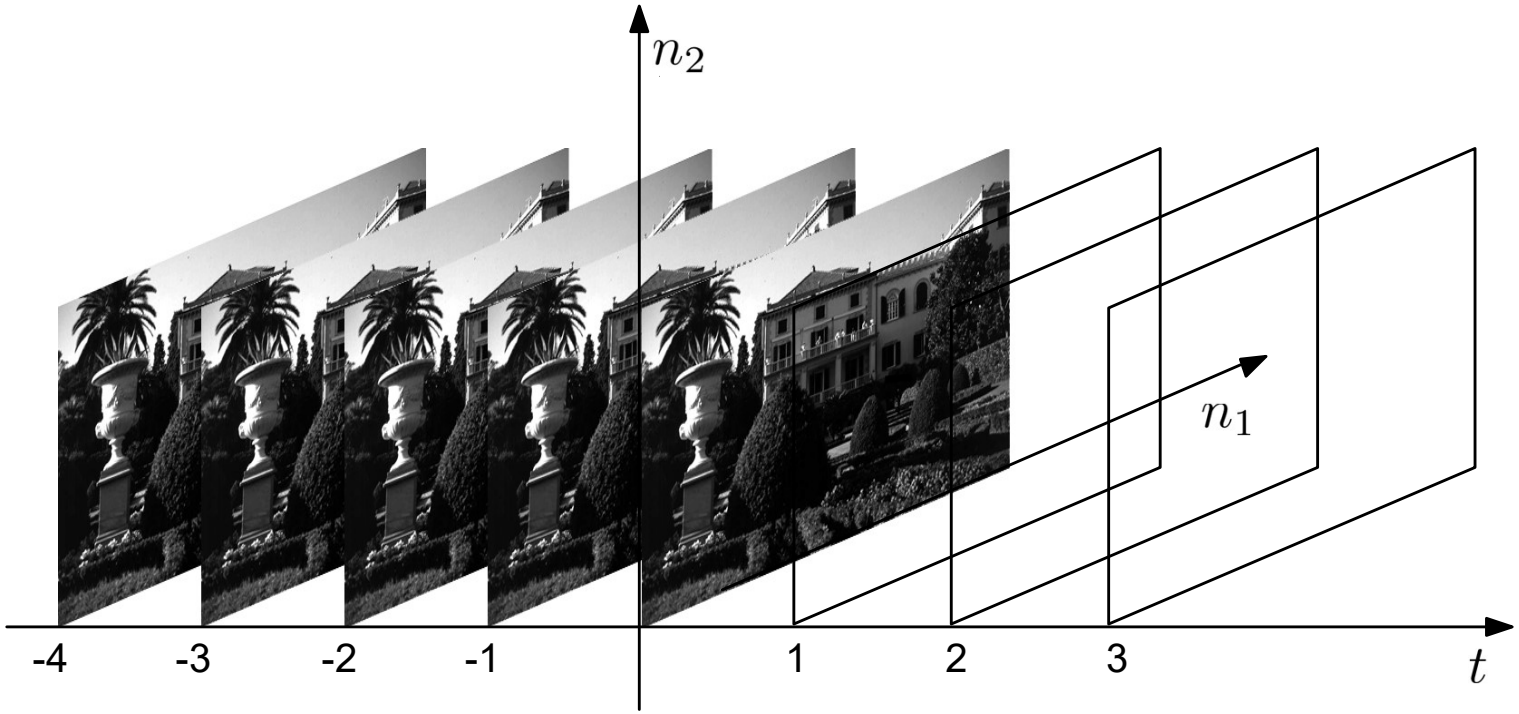
$$R_b = (16 \cdot 3 \cdot 3840 \cdot 2160 \cdot 120)\,\mathrm{b/s} \approx 47.78\,\mathrm{Gb/s}$$

Principles of video coding

We can compress a video stream, based on:

- spatial correlation, using the same principles exploited in image compressing
- temporal correlation, using time memory across successive frames

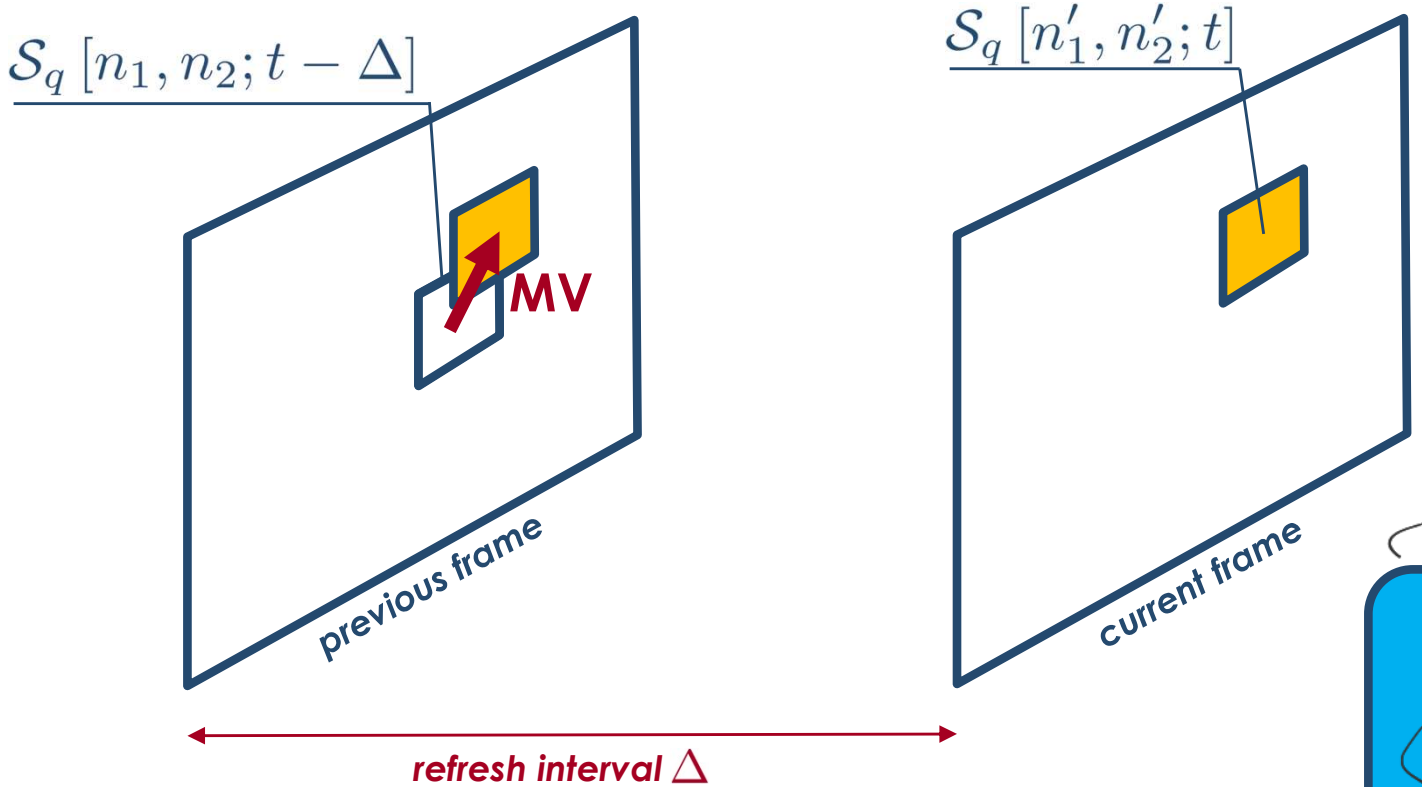
We can exploit this too!



Motion prediction (1/3)

Motion prediction (2/3)

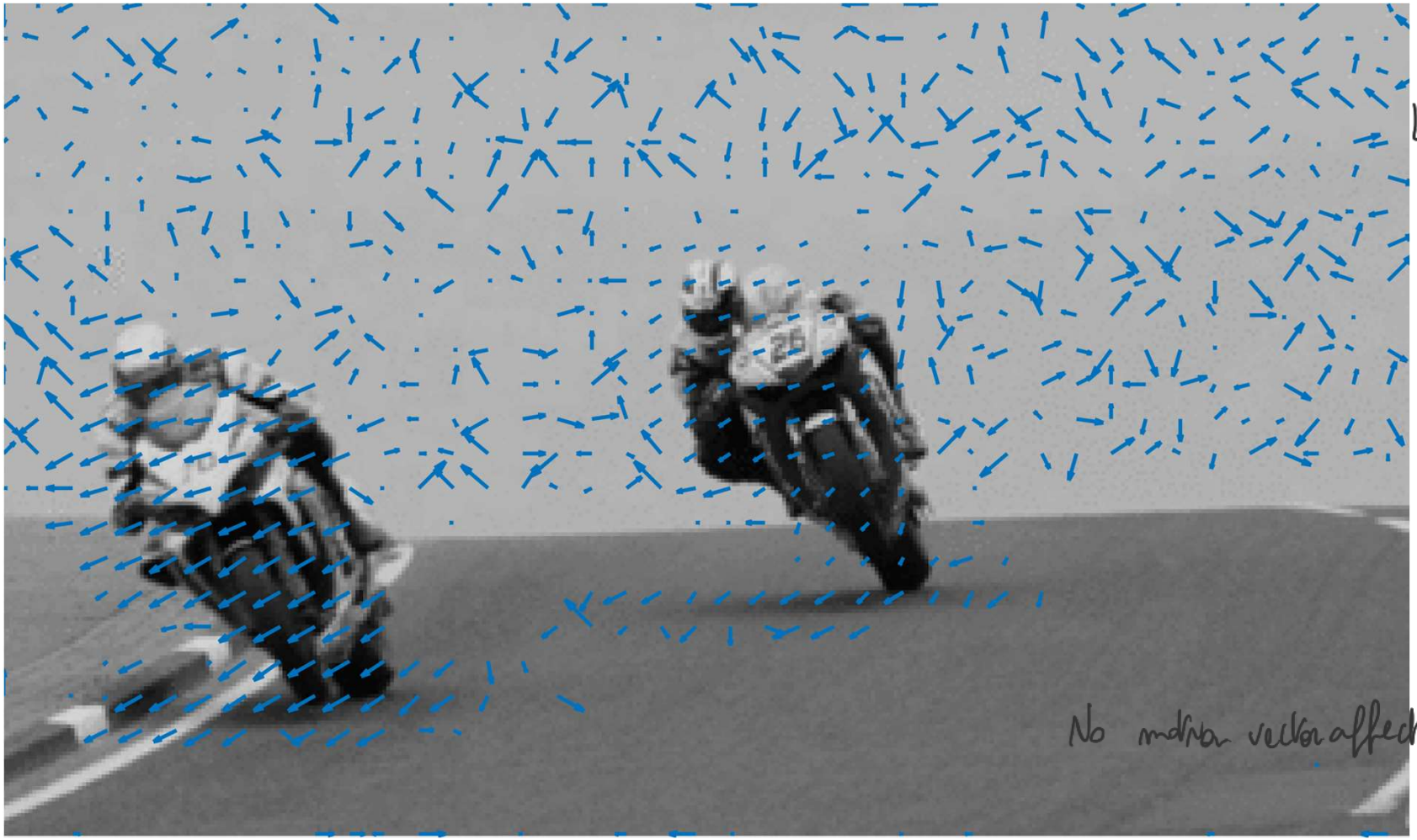
Effective video compression techniques rely on motion prediction based on motion vectors (MVs): Identify how parts of the frame has moved to the next frame. So instead of replicating the part we store a motion vector. MV can be translation, rotation, scaling factor. MV's not easy.



MPEG-1: 16×16 blocks
MPEG-2: 16×8 blocks
advanced standards: adaptive blocks

One approach: no exploit time correlation and encode every frame with JPEG. But I can save more with time correlation.

Motion prediction (3/3)

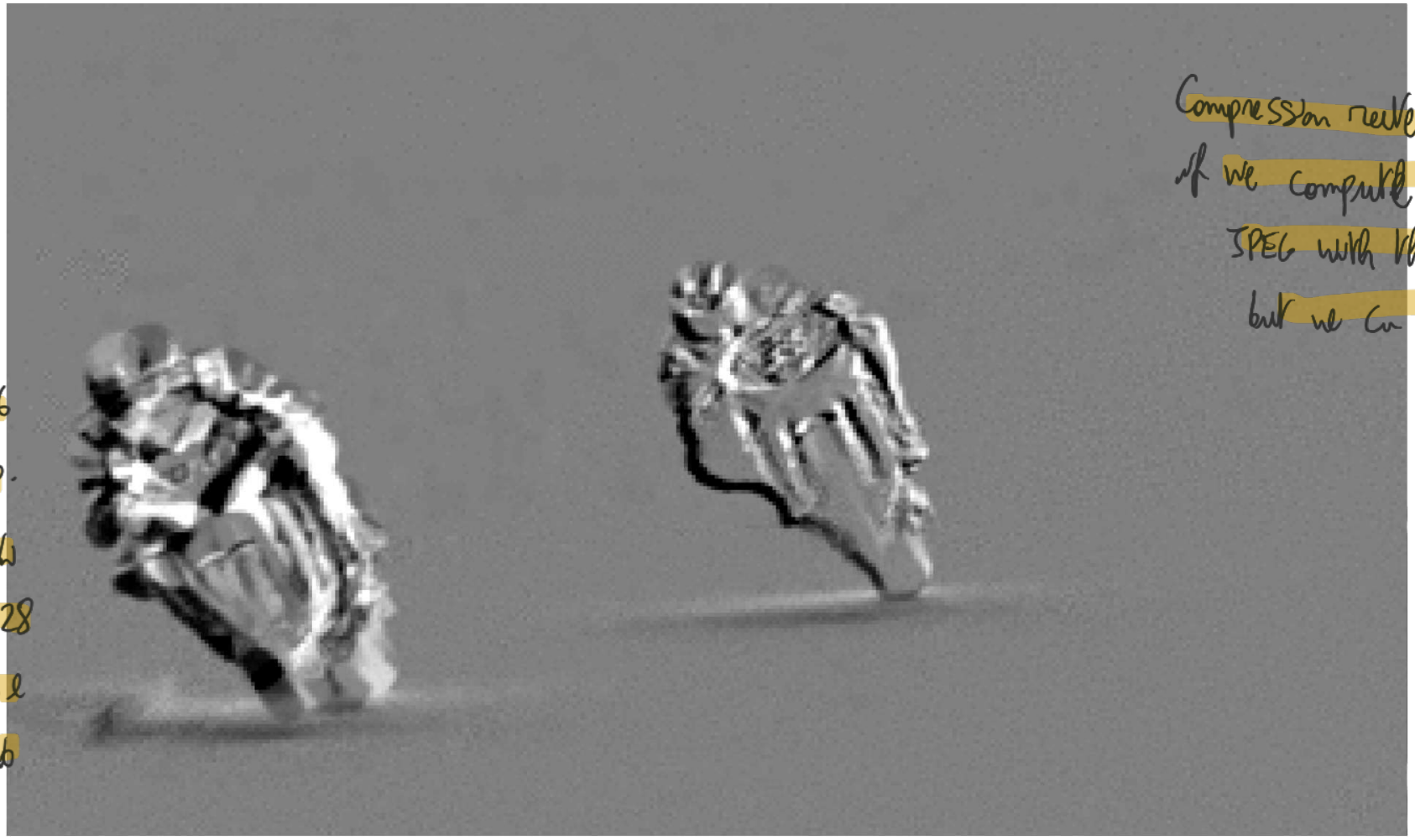


Motion compensation (1/2)

$$\mathcal{S}[n_1, n_2; t] - \mathcal{S}[n_1, n_2; t - \Delta]$$ **(without MC)**

Jpeg compress the difference

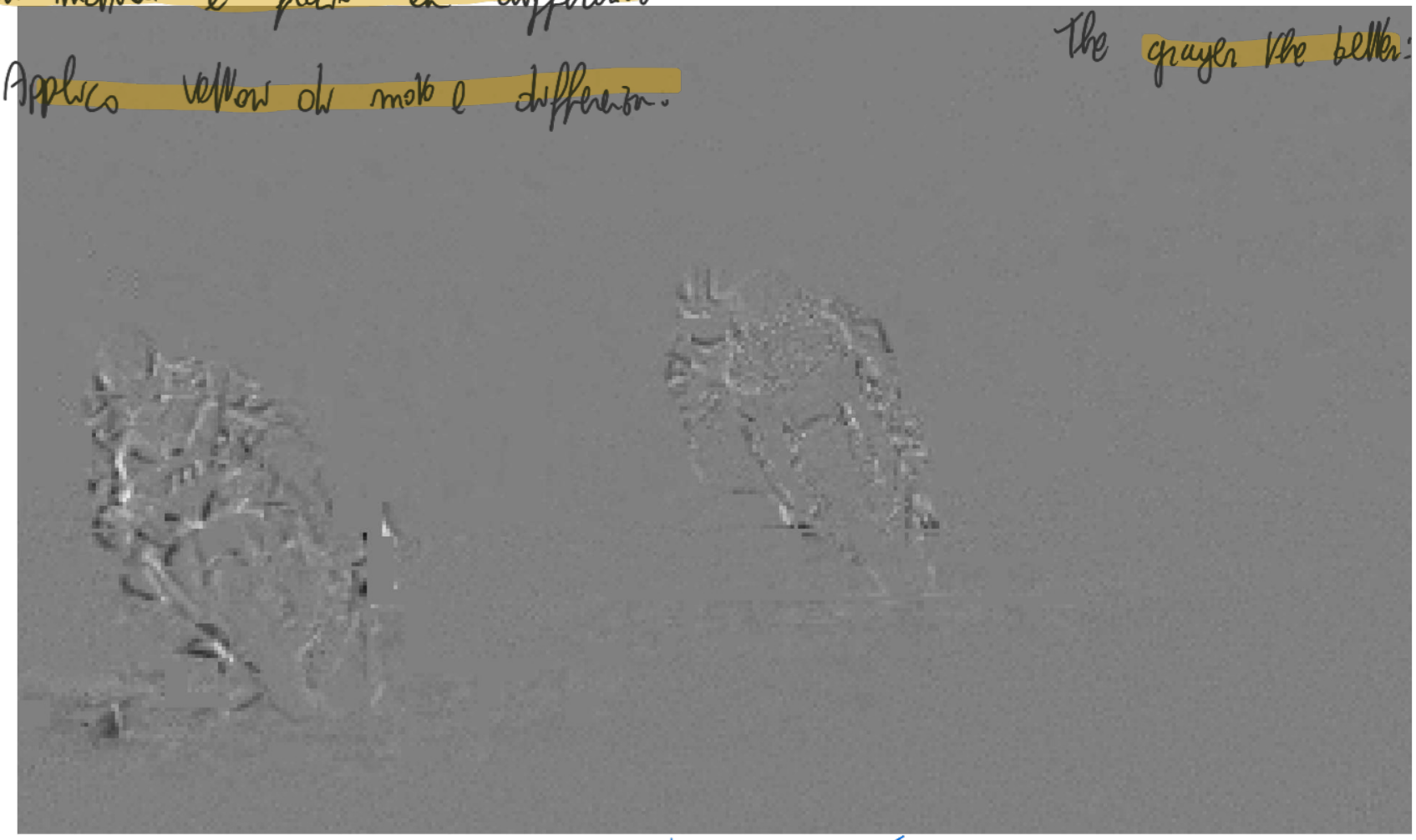
Compression results is better if we compute the JPEG with the difference but we can do better.



Diff
tra -256
a 256.
Rappresento
tra -128
a 128 e
rimuovo

Motion compensation (2/2)

$S[n_1, n_2; t] - \hat{S}[n_1, n_2; t]$ **(with MC)**

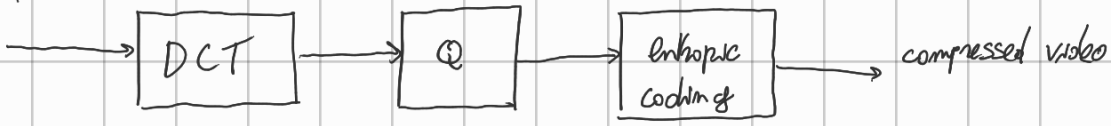


Dip. Ingegneria dell'Informazione
University of Pisa, Pisa, Italy

Giacomo Bacci

VIDEO

$S[m_1, m_2, t]$



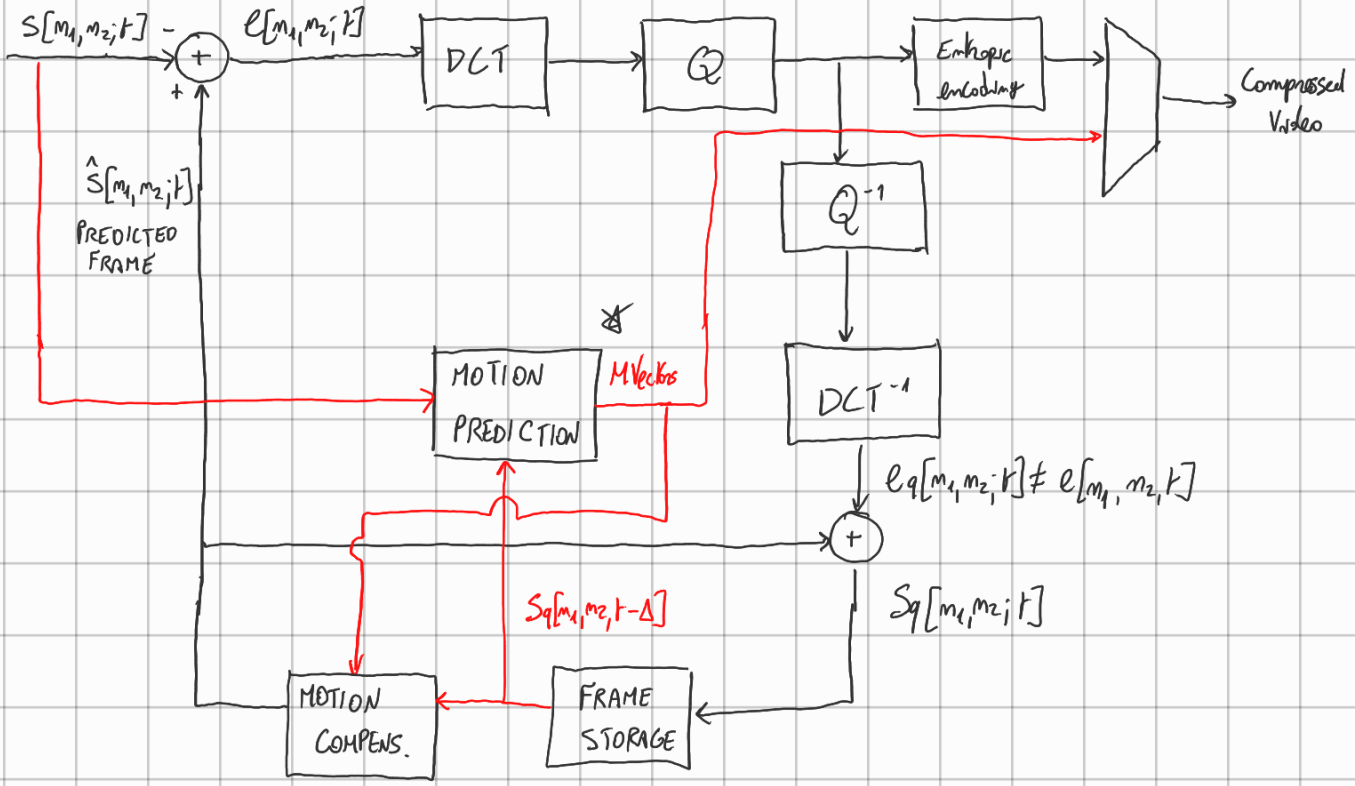
APPROACH TWO:

VIDEO

$S[m_1, m_2, t]$

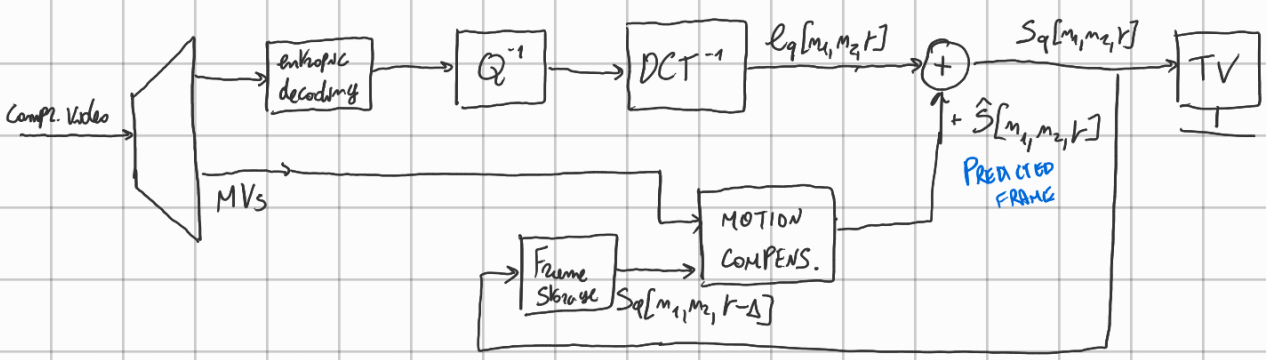
WE APPLY MOTION COMPENSATION

MPEG ENCODING



* This contains most of the complexity. Only at the encoder.

Only the encoder can do Mot. pred. because only it has the original stream.



There are also other tricks, we can exploit correlations between the past and the future. MPEG uses 4 blocks of frames.

The new algorithms use adaptive blocks. If 16x16 blocks are better, they are used.

In general, often some frames will be JPEG encoded frames. Every 12 there's a JPEG encoded one for a safety measure.

Other standards are better with motion prediction.

Examples of MPEG-2 compressed videos (1/3)

6 MB/s

4 MB/s

1.5 MB/s

Examples of MPEG-2 compressed videos (2/3)

6 MB/s

4 MB/s

1.5 MB/s

Examples of MPEG-2 compressed videos (3/3)

6 MB/s

4 MB/s

1.5 MB/s

Evolution of video compression standards (1/4)

The MPEG-2 standard (a.k.a. H.222/H.262) was designed to target standard definition (SD) used for DVD videos (first release: 1996)

With the increasing demand for additional quality, other standards have been deployed:

- advanced video coding (AVC), also referred to as H.264 or MPEG-4 Part 10 (first release: 2004), targets high definition (HD), thanks to adaptive macro-blocks, more effective entropic coding and lossy compression techniques
- high efficiency video coding (HEVC), also known as H.265 and MPEG-H Part 2 (first release: 2013), targets ultra-high-definition (UHD), thanks to improved MV prediction and motion compensation

Evolution of video compression standards (2/4)

MPEG-2: 3.54 MB

Evolution of video compression standards (3/4)

AVC: 1.19 MB

Evolution of video compression standards (4/4)

HEVC: 1.04 MB

Lossless source coding

Examples of lossless compression

Type of lossless comp. where input and compressed data have fixed lengths.

- **fixed-to-fixed code: ASCII code** take a fixed size of input sequence gives fixed size of the output sequence. $A = 65|_2 = 1000001$
- **fixed-to-variable code: Morse code** each input symbol is of fixed length, but the encoded output symbols can have varying lengths.
- **variable-to-fixed code: Lempel-Ziv-Welch algorithm** input symbols can have varying lengths but encoded output symbols are of fixed lengths.

start from a variable length and encode into a fixed amount of bits.

EX: I want to encode this sequence. Let's choose the fixed length of my encoder: M=5. When I choose M I need to build a lookup table with 2^{M-1} entries:

01101101001001010111011

Both encoder and decoder needs to know is the value of M and that the first two entries need to be 0 and 1.

index	input
0000	0
0001	1
0010	
0011	
0100	
⋮	
1111	

if M=5, $2^{M-1} = 16$ entries

We start scanning the sequence. 0. If is contained in the column of inputs. Keep scanning. 01. If is not in the lookup table. 1. Let's write 01 in the lookup table and we encode.

01101101111000

Is associated with the 0000 entry, so we substitute it.

00001

We keep scanning, 1, we have it, 0 then. We don't have 10. We repeat.

00001|0001|0

We have 1, but not 11. So...

Imagine:

00000000001

A B C

We have 0.

A

0000|0|0010|0|0011|0

index	input
0000	0
0001	1
0010	00
0011	000

Eventually you encode a lot of bits with just 5 bits.

If you run out of indexes there are tricks. If you have 0, 1 and 00 and 11 and 01 and 10, you can remove 0 and 1. If the compression rate goes down an escape sequence can be sent to reset the indexes.

How to decompress?

01101101111000

00001|00010|00011|00101|...

index	input
0000	0
0001	1
0010	01

The decoder just receives the encrypted sequence: 00001|00010|00011|00101|... with the agreement M=5. So we build the same matrix

0000 1 | 0001 0 | 0011 | 0010 1 | ...

index	input
0000	0
0001	1
0010	01
0011	10

other entries are empty.
Is this enough?

1. The decoder builds blocks of 5 bits. Take the first M-1 and decode them.

   0000 = 0, and then 1. Then we put the decoded word into the table

2. Repeat:

   0001 ⇒ 1, +0. We place 10 in 0011

   ⋮

4. 0010 = 01. + 1 ⇒ 011 new word.

An example of lossless compression: The ASCII code

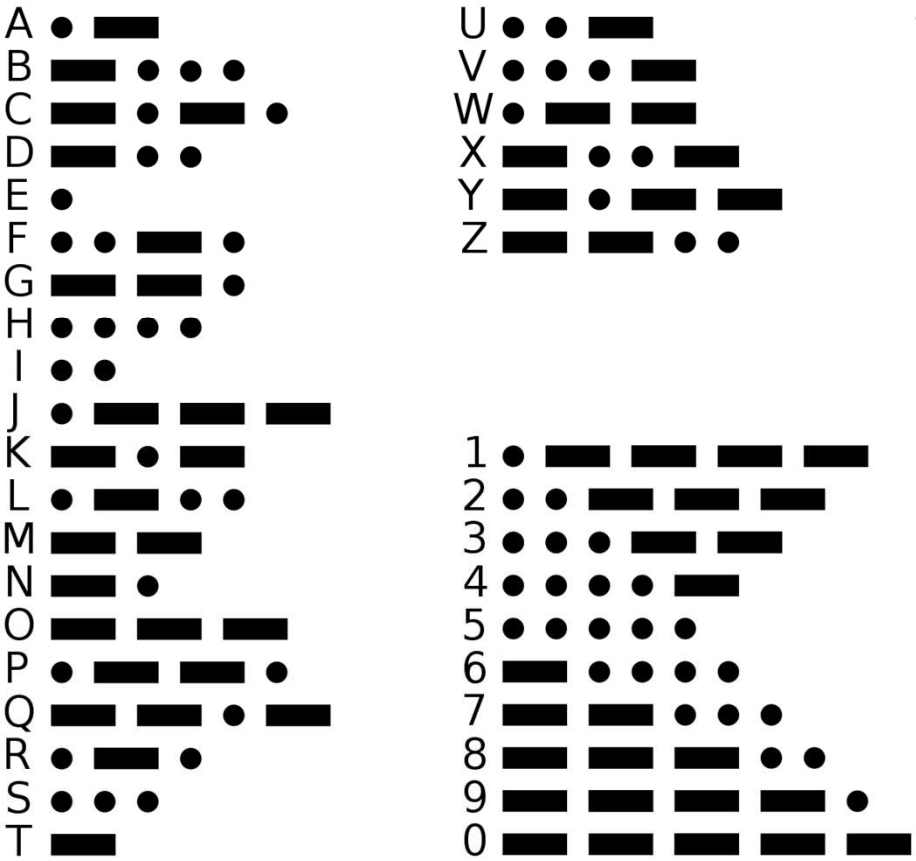
dec	hex	oct	char	dec	hex	oct	char	dec	hex	oct	char	dec	hex	oct	char
0	0	000	NULL	32	20	040	space	64	40	100	@	96	60	140	`
1	1	001	SOH	33	21	041	!	65	41	101	A	97	61	141	a
2	2	002	STX	34	22	042	"	66	42	102	B	98	62	142	b
3	3	003	ETX	35	23	043	#	67	43	103	C	99	63	143	c
4	4	004	EOT	36	24	044	$	68	44	104	D	100	64	144	d
5	5	005	ENQ	37	25	045	%	69	45	105	E	101	65	145	e
6	6	006	ACK	38	26	046	&	70	46	106	F	102	66	146	f
7	7	007	BEL	39	27	047	'	71	47	107	G	103	67	147	g
8	8	010	BS	40	28	050	(	72	48	110	H	104	68	150	h
9	9	011	TAB	41	29	051	)	73	49	111	I	105	69	151	i
10	a	012	LF	42	2a	052	*	74	4a	112	J	106	6a	152	j
11	b	013	VT	43	2b	053	+	75	4b	113	K	107	6b	153	k
12	c	014	FF	44	2c	054	,	76	4c	114	L	108	6c	154	l
13	d	015	CR	45	2d	055	-	77	4d	115	M	109	6d	155	m
14	e	016	SO	46	2e	056	.	78	4e	116	N	110	6e	156	n
15	f	017	SI	47	2f	057	/	79	4f	117	O	111	6f	157	o
16	10	020	DLE	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM	57	39	071	9	89	59	131	Y	121	79	171	y
26	1a	032	SUB	58	3a	072	:	90	5a	132	Z	122	7a	172	z
27	1b	033	ESC	59	3b	073	;	91	5b	133	[	123	7b	173	{
28	1c	034	FS	60	3c	074	<	92	5c	134	\	124	7c	174	|
29	1d	035	GS	61	3d	075	=	93	5d	135	]	125	7d	175	}
30	1e	036	RS	62	3e	076	>	94	5e	136	^	126	7e	176	~
31	1f	037	US	63	3f	077	?	95	5f	137	_	127	7f	177	DEL

An example of lossless compression: The Morse code

International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.

Short encoding with more likely letters.



The Lempel-Ziv-Welch (LZW) algorithm

- Lempel and Ziv patented the algorithm in 1978 as LZ78
- Welch further improved the algorithm in 1984
- easy and universal implementation, widely used for zip (and its variants) and GIF formats
- the original LZW algorithm encodes sequences as fixed-length 12-bit sequences

Encryption

Features of encrypted communications

- **Confidentiality**: third-party cannot read exchanged data
  attacks: eavesdropping, sniffing
- **Integrity**: third-party cannot change exchanged data
  attacks: man-in-the-middle
- **Authentication**: each party is sure who is really communicating with
  attacks: masquerading, spoofing, traffic generation

- **Availability**: time the system is in a functioning condition
  attacks: denial of service (DoS), distributed DoS (DDoS)

Channel coding

Channel coding (1/3)

Channel coding allows bit errors introduced by signal propagation through the error-prone channel to be either detected or corrected at the receive side

Coding embeds the signal constellation points in a higher dimensional signaling space than is needed for communications

Two ways of recovering the packets:

- forward error correction (FEC)
- automatic repeat request (ARQ)

$r = \frac{m}{K} \in [0,1]$. Called coding rate

If $r=1$, no channel encoding done.

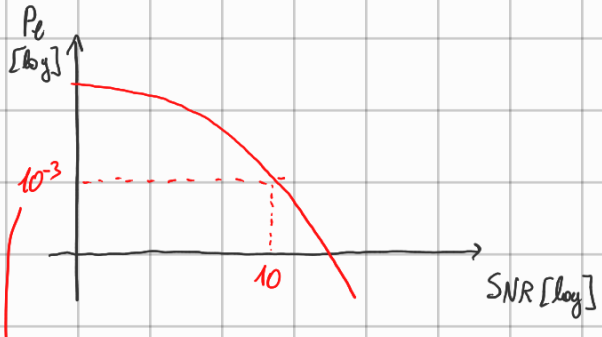
If r increases, not wasting resources. If r decreases, you are using more resources but you are protecting your signal.

Very basic channel encoding is using repetition codes. For every bit, K equal bits. So $r = \frac{1}{K}$.
When I send the signal, even with important noise, the probability of multiple errors is small. $P_e = P\{\text{more than } \frac{K-1}{2} \text{ errors}\}$ Note that K should be odd for better choices.

For this, one important ratio is $SNR = \frac{S}{N}$ (signal power / noise power)

What matters is the ratio, not the absolute value of those 2.

One metric we use is to watch the dependence between SNR and Prob. of errors:



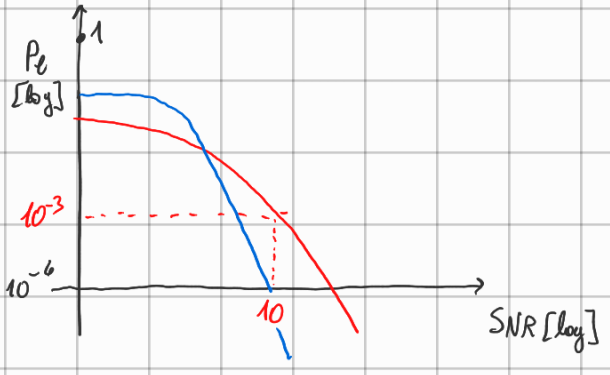
$SNR|_{dB} = 10 \log_{10}\left(\frac{S}{N}\right)$. If SNR increases, we get to a better situation.
The shape is like a fountain. Without channel coding we red.

With SNR=10, every 10^3 bits we have an error. For voice calls, BitErrorRate $= 10^{-2}$ works.
P_e = BER

Video: BER $= 10^{-5}$

file: BER $= 10^{-9}$ or 10^{-12} → QUASI ERROR FREE (QEF)

So suppose that you need to transfer a video which requires $BER = 10^{-5}$, but your hardware provides a $SNR = 10$, so a $BER = 10^{-3}$ (because of max power limitations for ex.). You can do channel coding, so we use 5 bits for 1. The typical performance will change like this:



With the same SNR you have a smaller BER, even lower than you need! But you need to send more data, so you need more bandwidth. Or you can use more power. It is a matter of trade offs: either you pay for performance or lower your target.

The math behind channel coding is information theory and there were a lot of progresses. Now channel coding is very close to the limit. We can use cheap hardware to achieve it.

MAIN CONCEPT: Performance can be measured in BER, function of SNR. For better performance, use more resources.

LEZIONE 7 END

Channel coding (2/3)

ARQ is not suitable for broadcast services, as retransmissions call for a return channel, which is not likely to be available

we need a feedback channel for the ACK. We can't always have one

ARQ happens at higher levels, so FEC is used at lower So we develop FEC

FEC reduces the need for retransmitting data, by including enough redundant information to permit the correction of errors at the destination

The coding rate is $r = R_b/R_c$

Transmitter and camera = receiver

We will have errors! There is channel coding there too

Dip. Ingegneria dell'Informazione
University of Pisa, Pisa, Italy

Giacomo Bacci

Channel coding* (3/3)

Between the FEC families:

Two main categories of FEC techniques:

- Block codes
- Convolutional codes

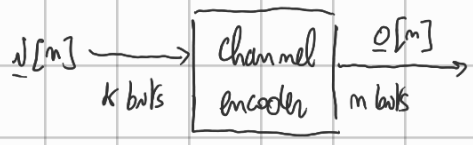
Block codes:

- All codewords have the same length
- Notable examples: Reed-Solomon (RS) and low-density parity check (LDPC) codes

Convolutional codes:

the out depends both on the current sequence but also on the past of the encoding.

- Codewords depend on data messages and a given number of previously encoded messages
- Notable examples: Turbo codes



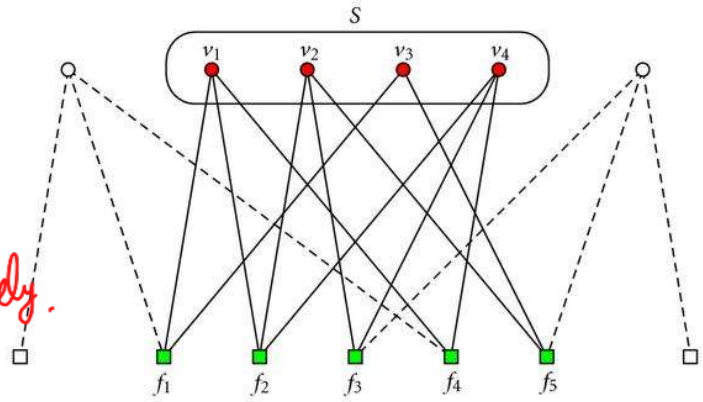
Given an input sequence, you get an output sequence with fixed dimension thanks to the matrix.

$$\underline{v}[m] = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\underline{o}[m] = \underline{\underline{H}}^{m \times k} \cdot \underline{v}[m]$$

An example of block codes: LDPC codes*

- LDPC codes are linear block codes characterized by sparse parity check matrices



- Invented by Gallagher in 1960, but reproposed by MacKay and Neal in 1995, when technology of iterative decoding was mature enough
- Particularly performing with large block lengths (e.g., 64,800 bits)
- LDPC codes yield QEF reception even in poor link conditions

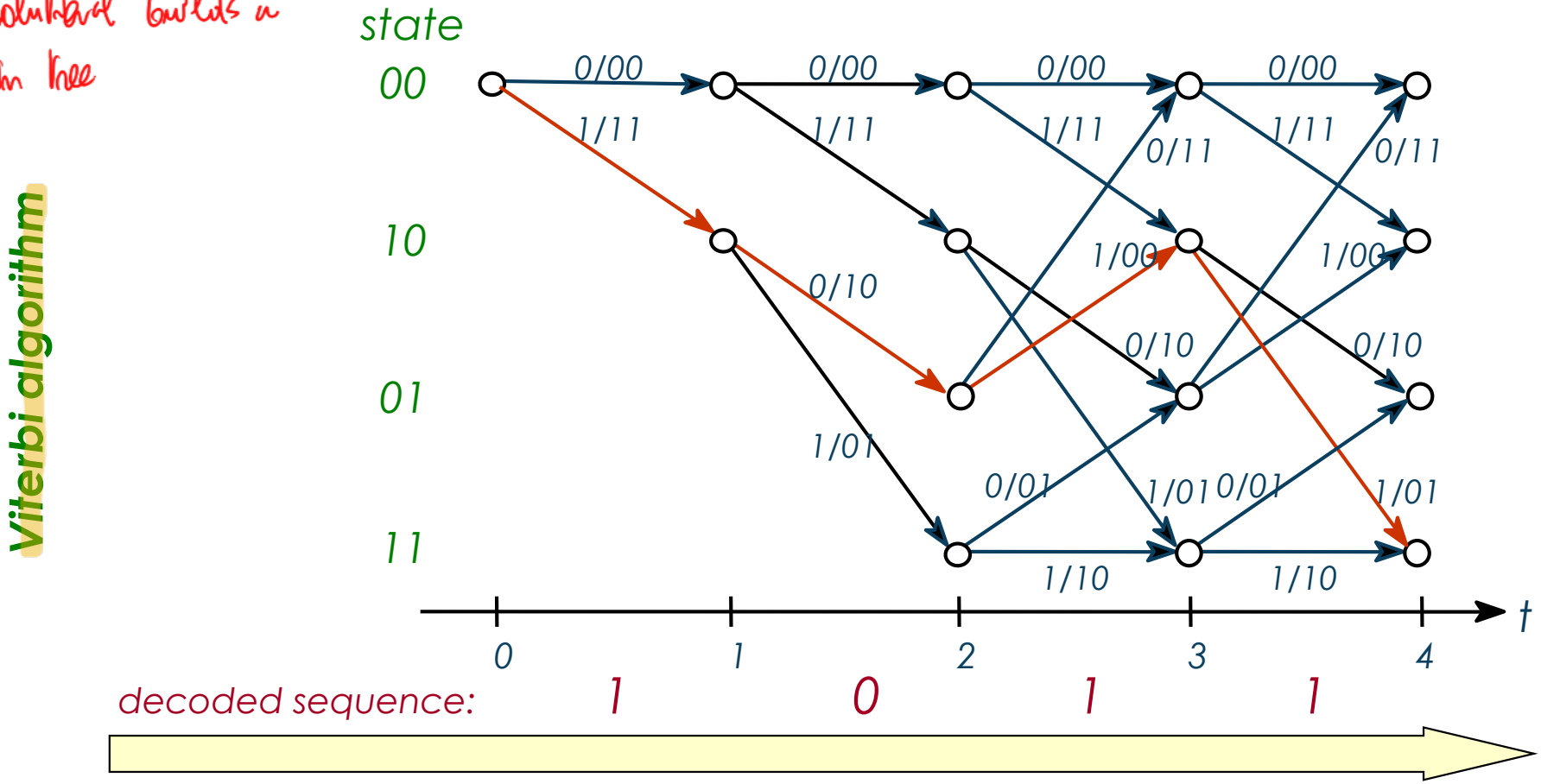
Dip. Ingegneria dell'Informazione
University of Pisa, Pisa, Italy

Giacomo Bacci

An example of convolutional codes*

Concatenated encoding yields exponentially decreasing error probability with polynomial-time decoding complexity:

Convolutional encoder builds a certain tree



Modulation

We need to transform the dig. sig. into the analog domain.

Why modulate?

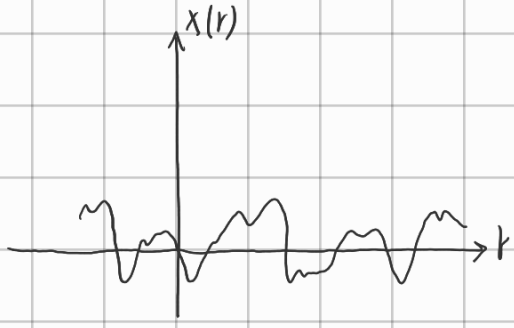
$\lambda = \frac{c}{f}$

- baseband is not sustainable, in terms of physical feasibility of the components
- bandpass yields additional benefits, e.g., in terms of multiple access

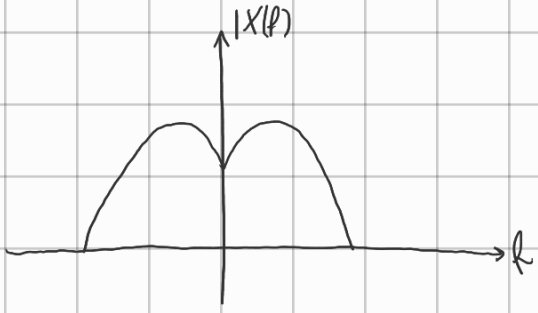
1. Turn digital to analog
2. The frequency should allow us to build physical antennas
3. Many user need to share the same medium.

Playing with the frequencies helps with multiple access.

BASEBAND Take $X(t)$ analog signal. Let's transform it into the realm of frequencies.



$$X(f) = \int_{-\infty}^{+\infty} x(t) e^{-2\pi j f t} dt$$



If we work around $f=0$, we have a Fourier transform of a BASEBAND SIGNAL. (a BS is a signal whose spectrum is around 0, or the shape is low pass, so important frequencies around 0).

I would like to move the spectrum around a carrier frequency of f_0. And of course the spectrum will still need to be simmetric. That's called a BANDPASS SIGNAL: because I don't have sig. comps around 0 and it has bandpass shape.
To do that we can use the theory of FT.

Turns out, if I build $y(t) = x(t) 2\cos(2\pi f_0 t)$ I get the solution. THIS IS THE MODULATION TH.

$2\cos(2\pi f_0 t) = e^{j 2\pi f_0} + e^{-j 2\pi f_0}$ $\Rightarrow$ So the transform only adds a delay.

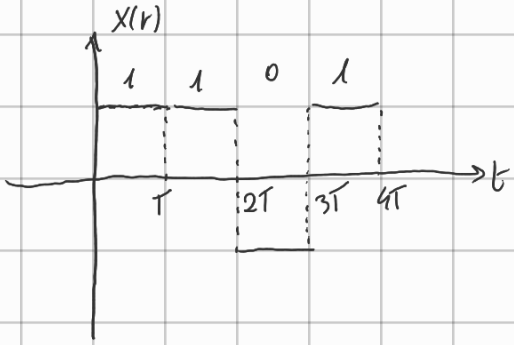
$$Y(f) = X(f-f_0) + X(f+f_0)$$

THIS UNDER THE ASSUMPTION THAT $f_0 \gg B$

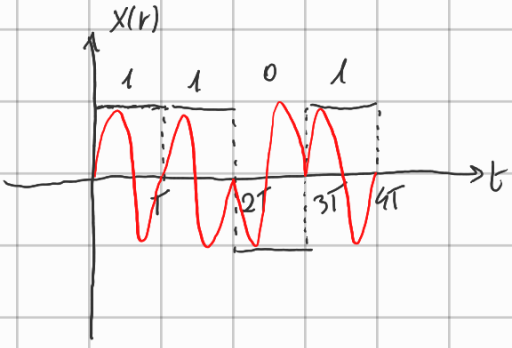
How do we get an analog signal from the bits?
I can build a mapping system that maps bits into symbols in the real domain.

Let's take the complex plane. For 1 I assign the value 1. For 0 the value -1. Then I multiply my bits for a pulse:

$$X(t) = \sum_k a_k \, p(t - T)$$



So we can do this but we have a problem. The info is contained in the amplitude of the signal. In general this is not a good choice, because the cheapest amplifiers are not linear so you cannot use them. The trick is to send info in the phase of the signal: if I send a +1 I send:



Now I can use non linear amplifiers. What matters is that the phase transition is maintained.

This is Phase shift keying (PSK). The shift in the phase is the modulation. Here this is called BPSK, binary PSK.

So after we have the signal we move it at the right frequency.

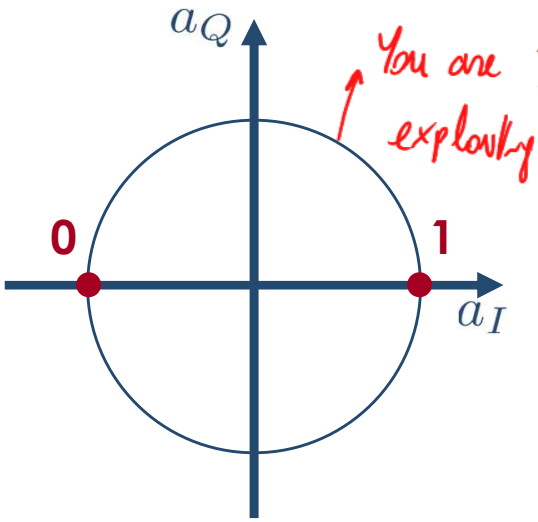
MAPPING: transform a bit into a baseband signal and then I modulate it.

NOTE: Here $p(t-T)$, the shaping pulse is the same waveform we use to represent a bit.

PSK and QAM constellations (1/2)

The input streams can be mapped in the I-Q plan, based on different criteria:

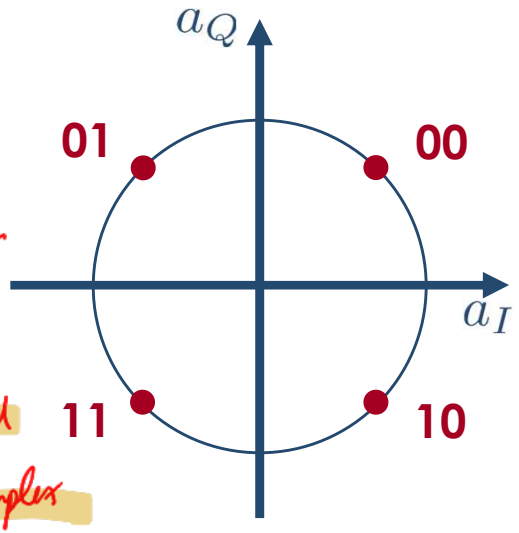
BPSK:



You are just exploiting the real axis

Input	I	Q
0	-1	0
1	+1	0

QPSK:



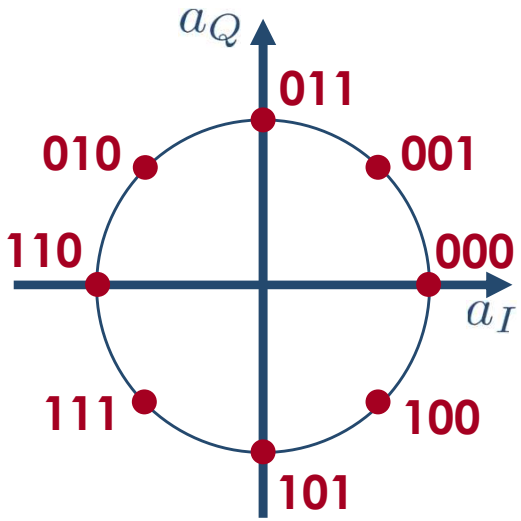
But in baseband you can use complex signals.

Input	I	Q
00	$+2^{-1/2}$	$+2^{-1/2}$
01	$-2^{-1/2}$	$+2^{-1/2}$
10	$+2^{-1/2}$	$-2^{-1/2}$
11	$-2^{-1/2}$	$-2^{-1/2}$

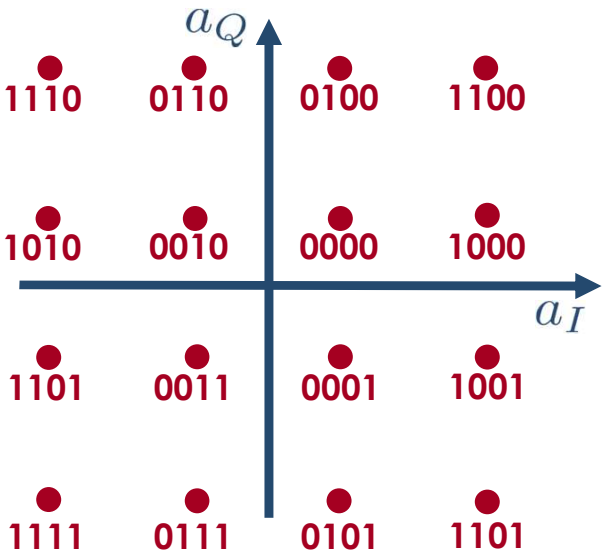
PSK and QAM constellations (2/2)

The input streams can be mapped in the I-Q plan, based on different criteria:

8PSK:



16-QAM:



Input	I	Q	Input	I	Q
000	+1	0	100	$+2^{-1/2}$	$-2^{-1/2}$
001	$+2^{-1/2}$	$+2^{-1/2}$	101	0	-1
010	$-2^{-1/2}$	$+2^{-1/2}$	110	-1	0
011	0	+1	111	$-2^{-1/2}$	$-2^{-1/2}$

In-phase (I) – in-quadrature (Q) modulation

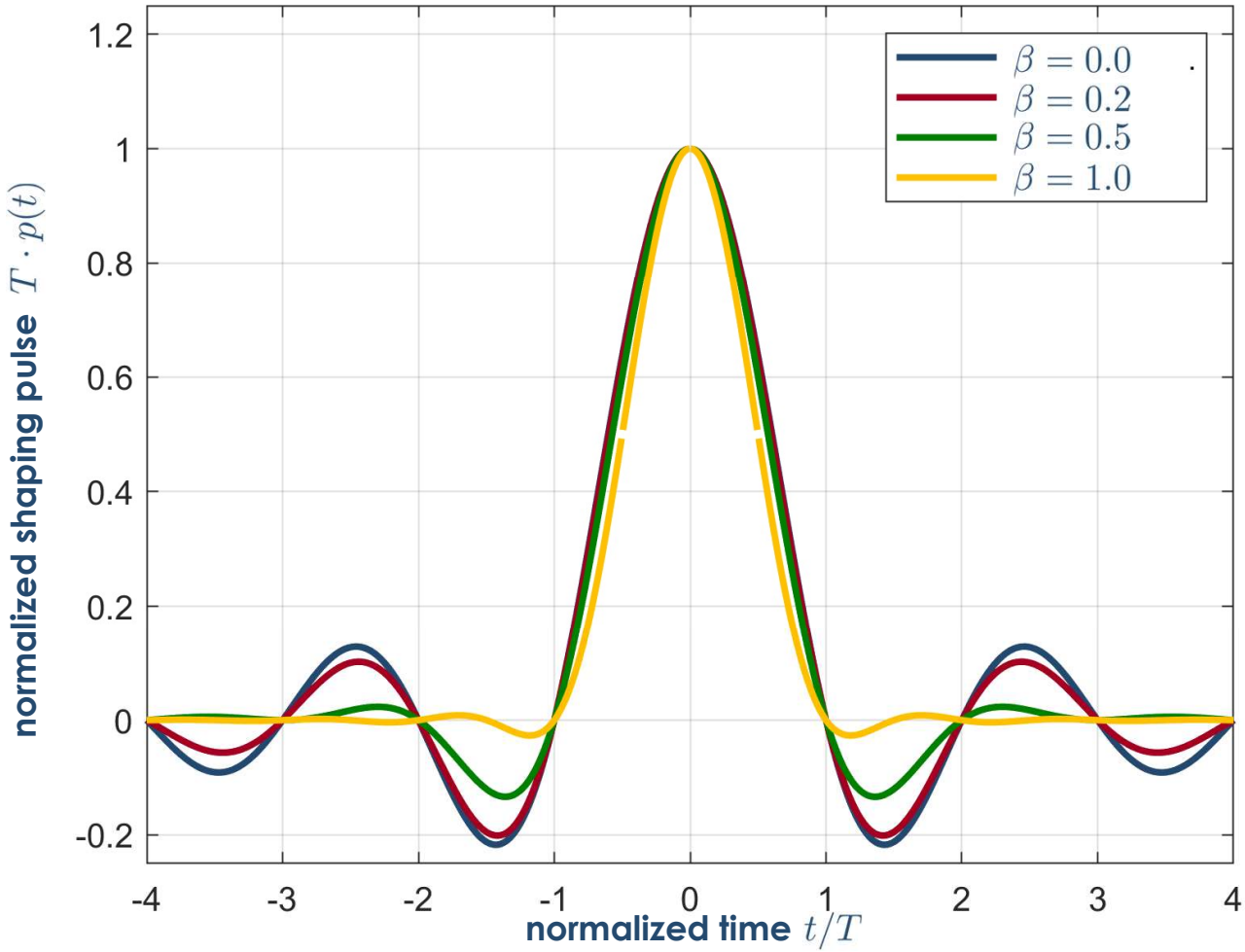
- we can take advantage of the I-Q modulation scheme to **convey** a digital signal through an analog carrier

- pulse-coded modulations have **infinite** bandwidth: to limit the bandwidth while (re)converting it to the analog domain, we can process the binary digits with a (low-pass) **shaping pulse** $p(t)$

Shaping pulse (1/2)

A widely used shaping pulse is the square root raised cosine (SRRC) waveform:

BITS

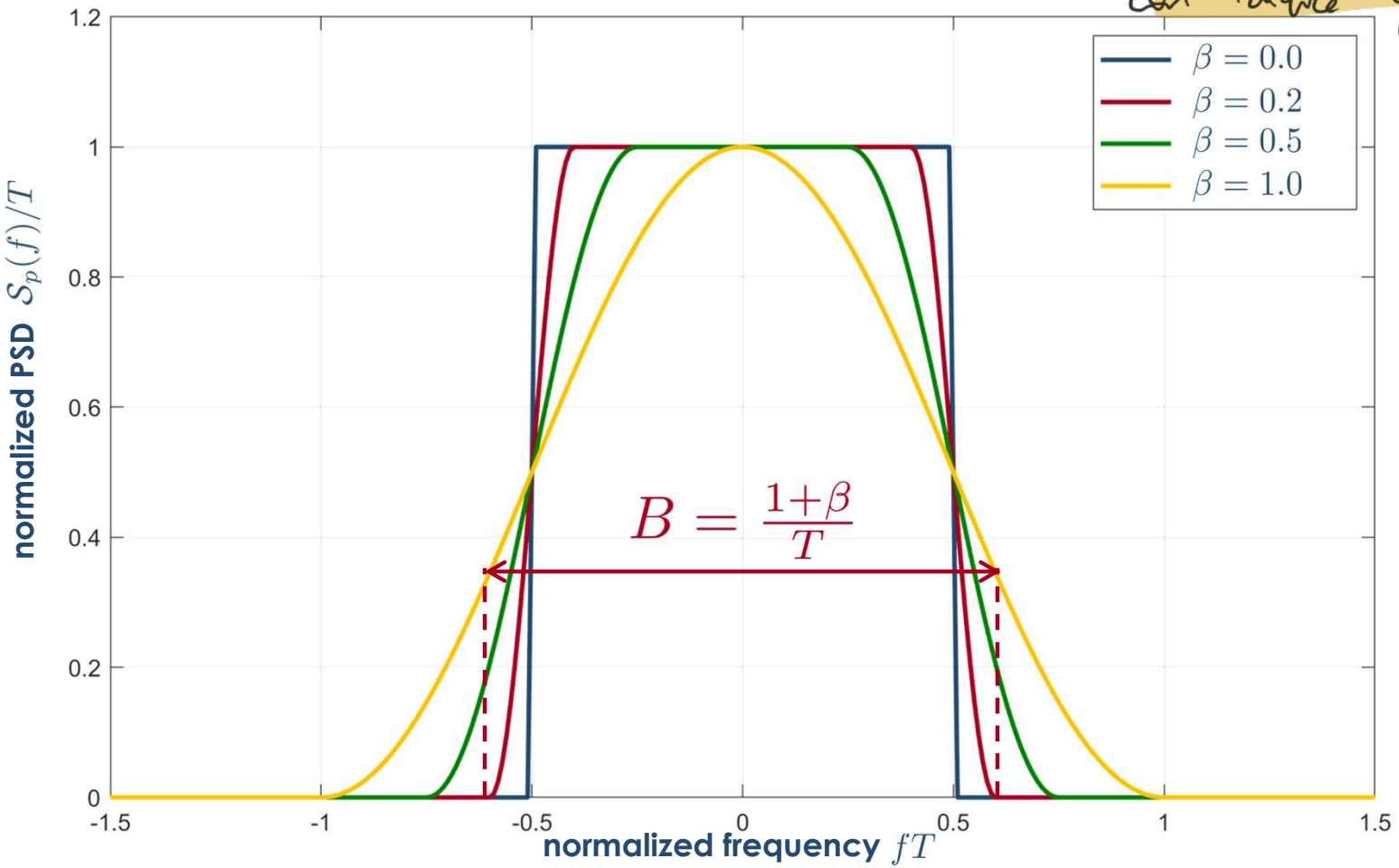


Dip. Ingegneria dell'Informazione
University of Pisa, Pisa, Italy

Shaping pulse (2/2)

A widely used shaping pulse is the **square root raised cosine (SRRC)** waveform:

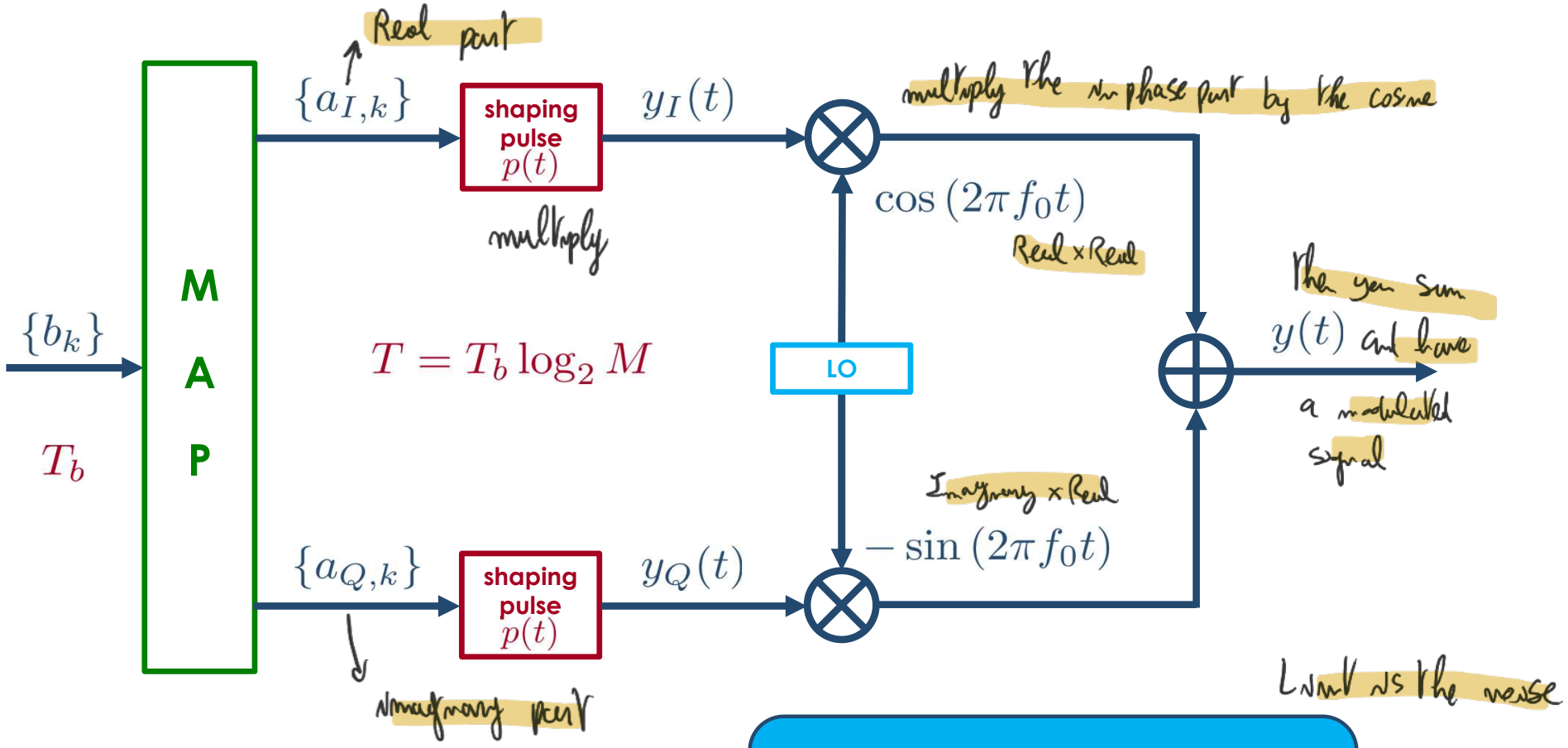
Forma d'onda a coseno con radice quadrata



Dip. Ingegneria dell'Informazione
University of Pisa, Pisa, Italy

Giacomo Bacci

The I-Q modulator



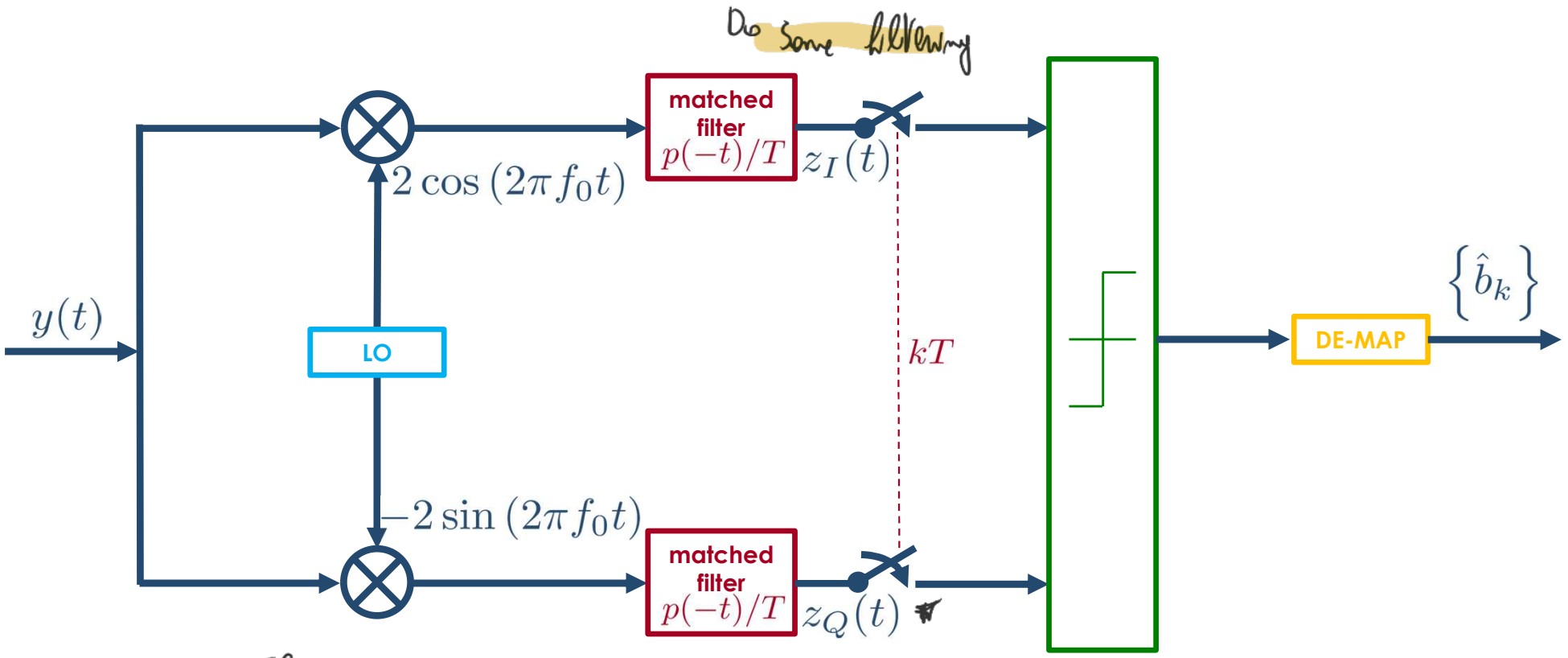
General form of a bandpass signal*

$$\cos(\alpha+\beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta$$

$$\begin{aligned} y(t) &= x(t)\cdot\cos\left[2\pi f_0 t + \vartheta(t)\right] \\ &= \underbrace{x(t)\cos\left[\vartheta(t)\right]}_{y_I(t)}\cdot\cos(2\pi f_0 t) \\ &- \underbrace{x(t)\sin\left[\vartheta(t)\right]}_{y_Q(t)}\cdot\sin(2\pi f_0 t) \end{aligned}$$

this is valid under the hypothesis that $y_I(t)$ and $y_Q(t)$ vary slowly with respect to carrier f_0

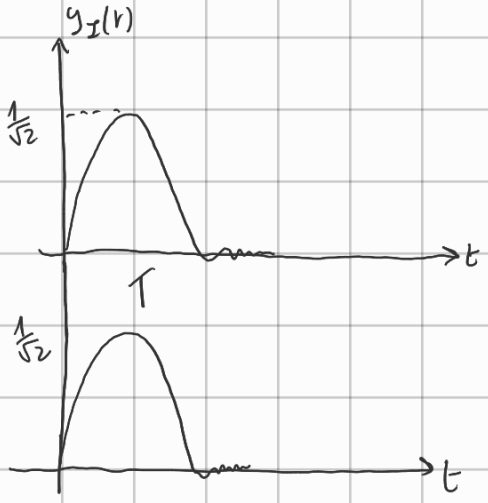
The I-Q demodulator (1/2)



$y(t) = x(t) 2\cos(2\pi f_0 t)$ How to demodulate?

$$\mathcal{F}\{y(t) \cdot 2\cos(2\pi f_0 t)\} = Y(f-f_0) + Y(f+f_0) = X(f-2f_0) + X(f+2f_0) + 2X(f)$$

In the example of 00, 01, 10, 11: for a_I in 00 I have $+\frac{1}{\sqrt{2}}$, and $+\frac{1}{\sqrt{2}}$ for a_Q



00 JMP BITS

Idea per il demapping: sulla base del quadrante, scegli la decodifica.

LEZIONE 8 END

The I-Q demodulator* (2/2)

$$2\cos^2\alpha = 1 + \cos 2\alpha,\ f_0 \gg B$$

$$z_I(t) = [2y(t)\cos(2\pi f_0 t)] \otimes p(-t) \approx y_I(t)$$

$$z_Q(t) = [-2y(t)\sin(2\pi f_0 t)] \otimes p(-t) \approx y_Q(t)$$

$$2\sin^2\alpha = 1 - \cos 2\alpha,\ f_0 \gg B$$

Signal-to-noise ratio (SNR)

The main figure of merit becomes the signal-to-noise ratio (SNR), in the form of

$$\frac{E_b}{N_0} = \frac{S}{N} \cdot \frac{B}{R_b}$$

- E_b: energy per bit
- N_0: noise power spectral density
- S: useful signal power
- N: noise power
- B: signal bandwidth
- R_b: bit rate

S = signal power

N = noise power

(Power = Energy / Time)

So S = Energy of each bit / Bit time

$= E_b \cdot R_b$

The performance of a digital communication system can be assessed measuring the BER – which occurs when $\hat{b}_k \neq b_k$ – and the SER – which occurs when $(\hat{a}_{I,k} + \hat{a}_{Q,k}) \neq (a_{I,k} + a_{Q,k})$

Dip. Ingegneria dell'Informazione
University of Pisa, Pisa, Italy

Giacomo Bacci

$N = B \cdot N_0$

- N_0 → Measure of energy associated to noise.
- B → Bandwidth of the system

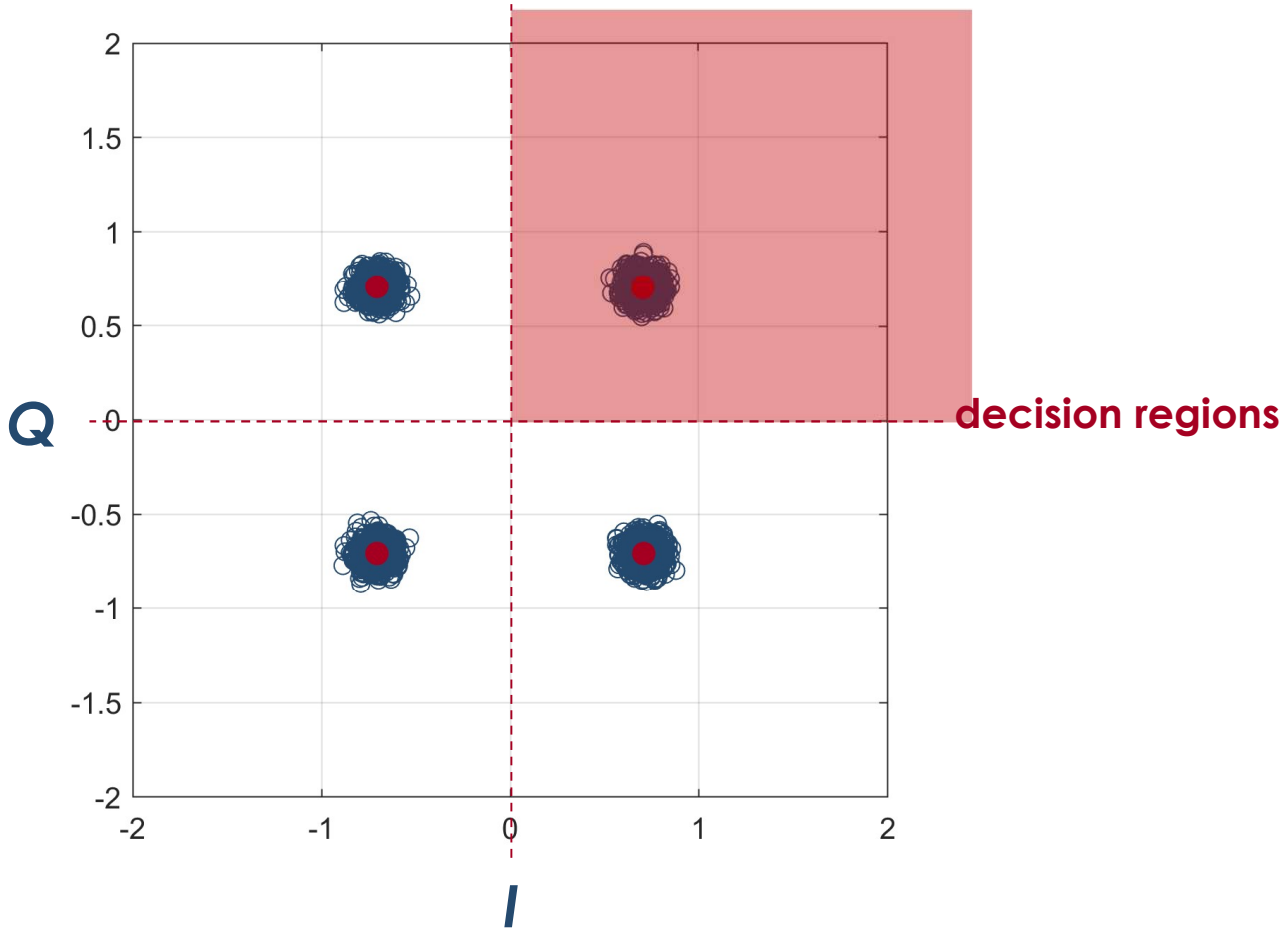
$$\frac{S}{N} = \frac{E_b R_b}{N_0 B} = \frac{E_b}{N_0} \cdot \frac{R_b}{B}$$

- $\frac{R_b}{B}$ → Spectrum efficiency: how well the system exploits the bandwidth.
- $\frac{E_b}{N_0}$ → Measure of the energy efficiency: how good the system deploys energy.

Impact of additive white Gaussian noise (AWGN) (1/5)

QPSK @

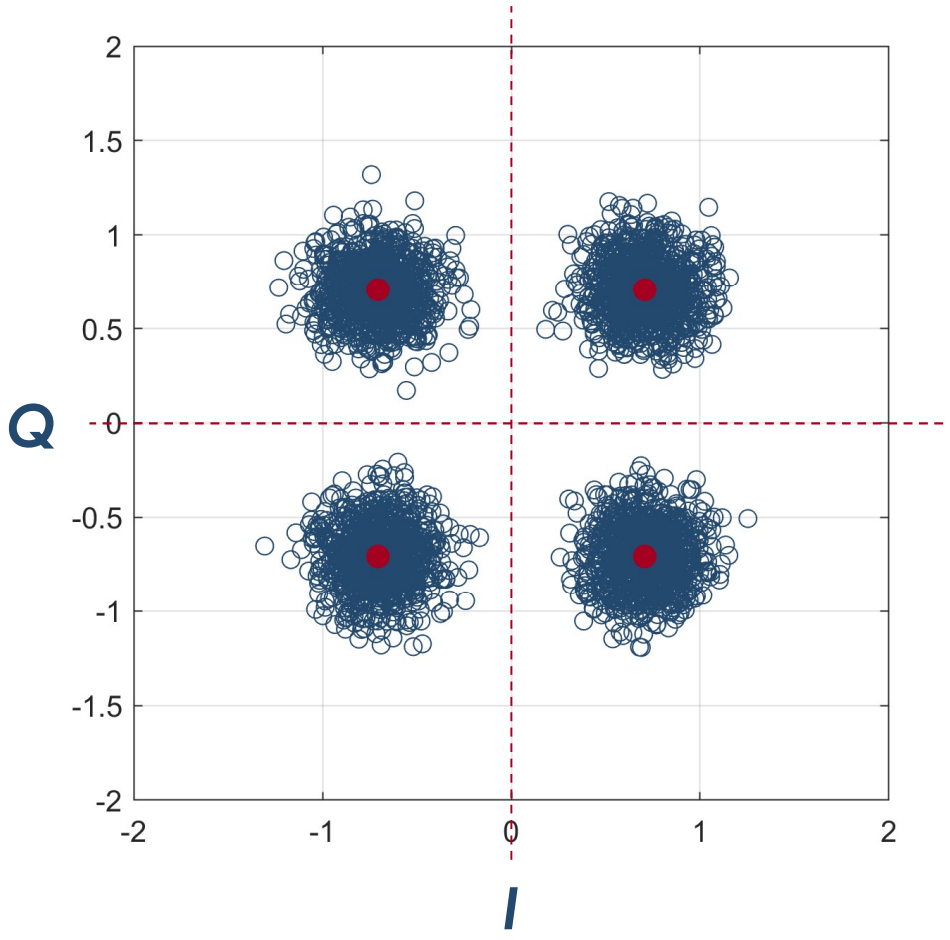
$E_b/N_0 = 20\,\mathrm{dB}$



Impact of additive white Gaussian noise (AWGN) (2/5)

QPSK @

$E_b/N_0 = 10\,\mathrm{dB}$

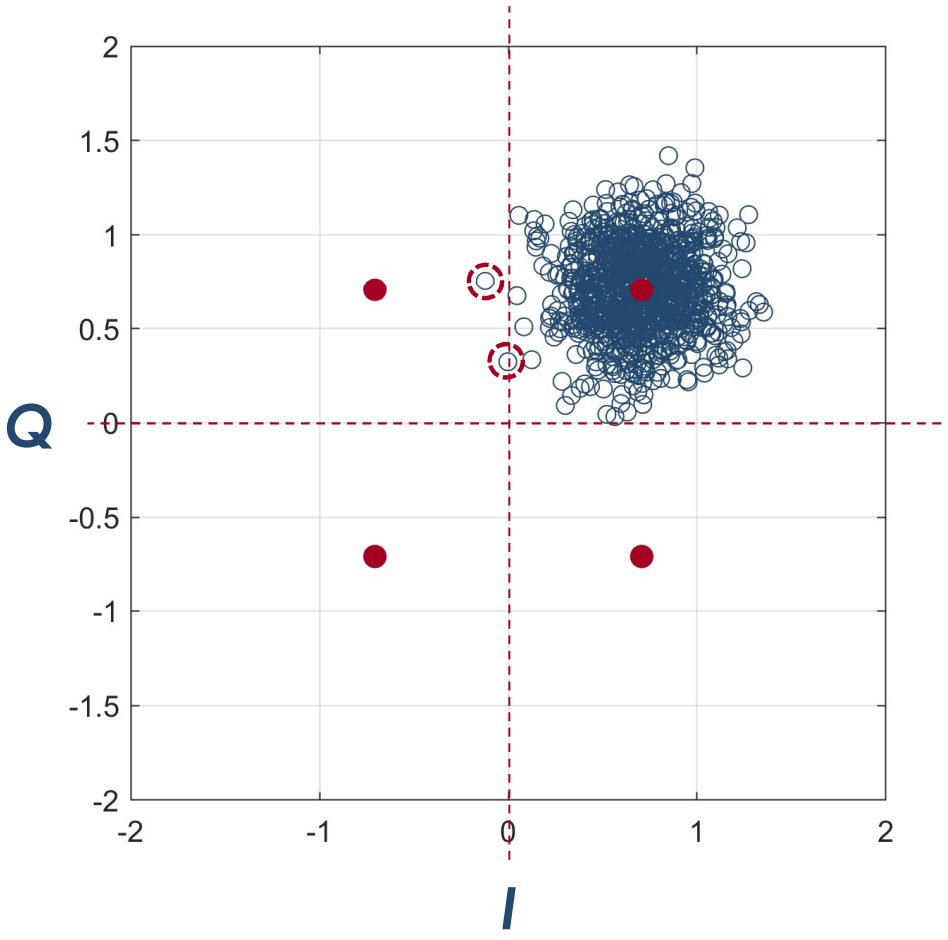


Impact of additive white Gaussian noise (AWGN) (3/5)

When introducing AWGN, errors may arise!

QPSK @

$E_b/N_0 = 7\,\mathrm{dB}$

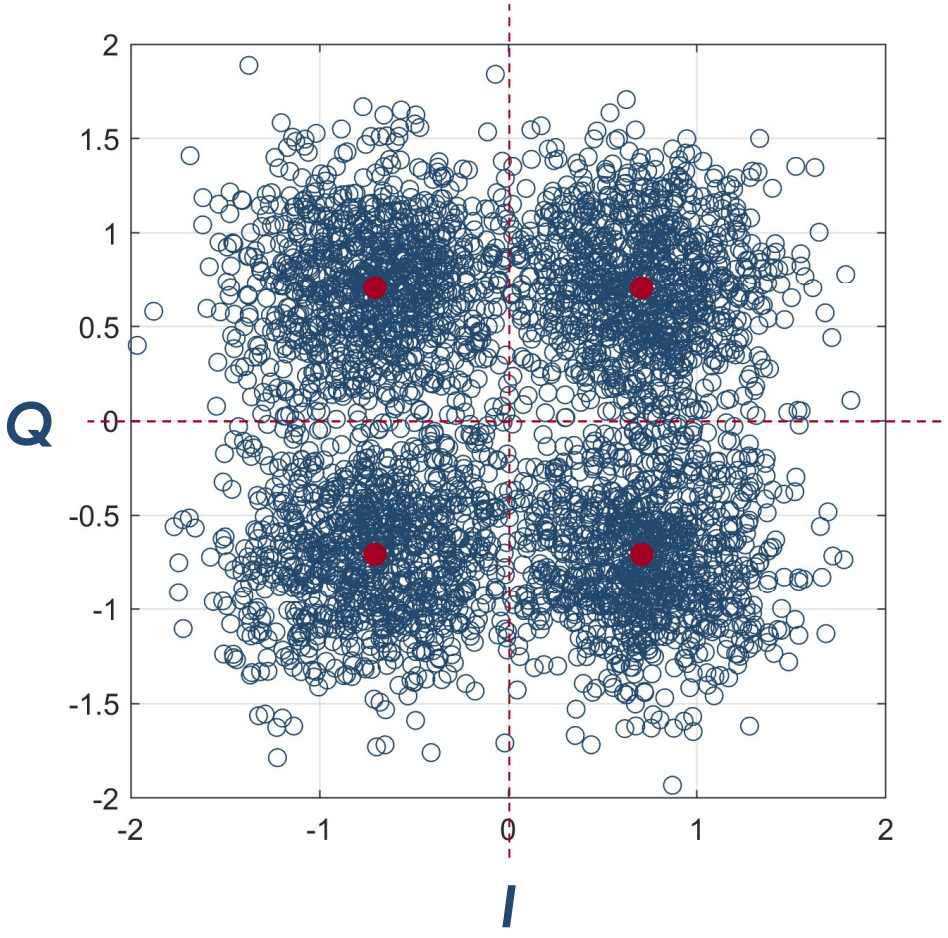


Impact of additive white Gaussian noise (AWGN) (4/5)

The situation becomes dramatic when increasing the AWGN…

QPSK @

$E_b/N_0 = 3\,\mathrm{dB}$

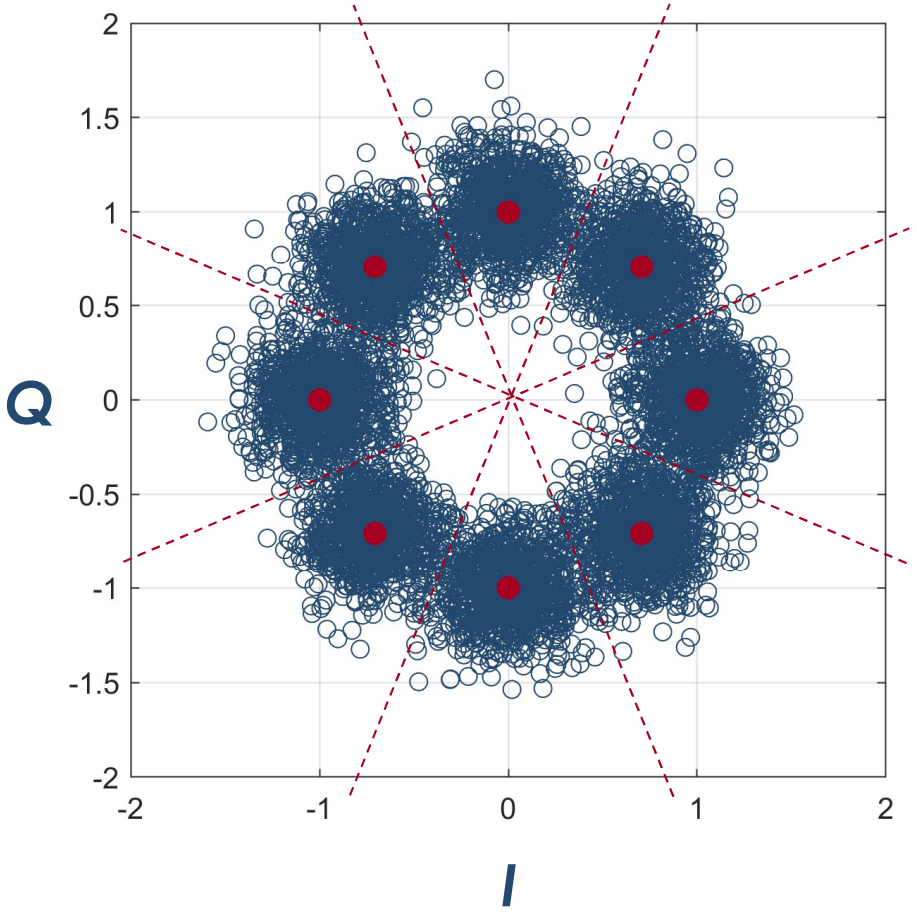


Impact of additive white Gaussian noise (AWGN) (5/5)

…or when increasing the modulation order

8PSK @

$E_b/N_0 = 7\,\mathrm{dB}$



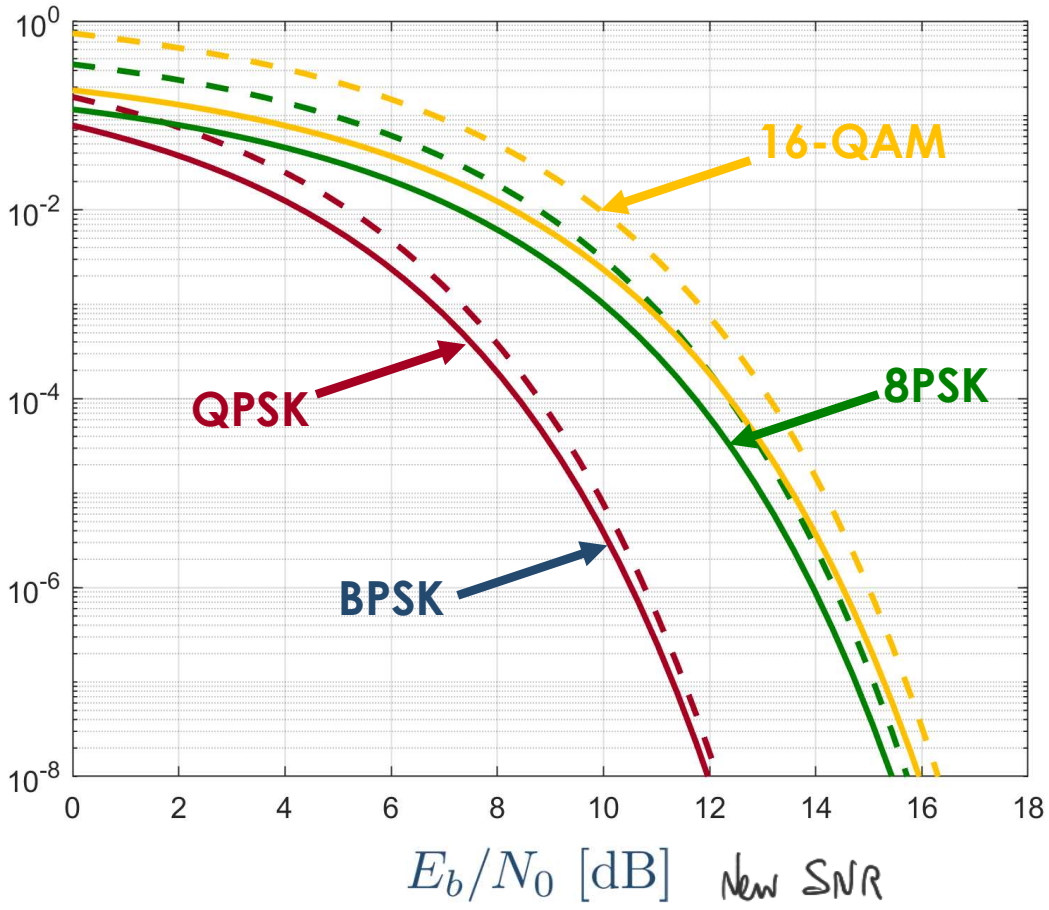
Bit error rate (BER) and symbol error rate (SER) (1/2)

The main figure of merit becomes the signal-to-noise ratio (SNR), in the form of

$$\frac{E_b}{N_0} = \frac{S}{N} \cdot \frac{B}{R_b}$$

The performance of a digital communication system can be assessed measuring the BER – which occurs when $\hat{b}_k \neq b_k$ – and the SER – which occurs when $(\hat{a}_{I,k} + \hat{a}_{Q,k}) \neq (a_{I,k} + a_{Q,k})$

Bit error rate (BER) and symbol error rate (SER) (2/2)



—— BER

------- SER

To achieve a 10^{-5} BER with QPSK, you need a $\frac{E_b}{N_0} \approx 10$ dB

To get better you need to increase the energy carried by the bits, so you need to give more power.

Taking more bits per symbol increases the spectrum efficiency:

$$\frac{S}{N} = \frac{E_b}{N_0} \cdot \frac{R_b}{B}$$

↳ The performance gets worse though. The impact of the noise is becoming more important. To improve it you should increase the energy of the bit. With an even bigger budget you could use 16 QAM. If the noise is fixed, to increase E_b/N_0 you need more power.

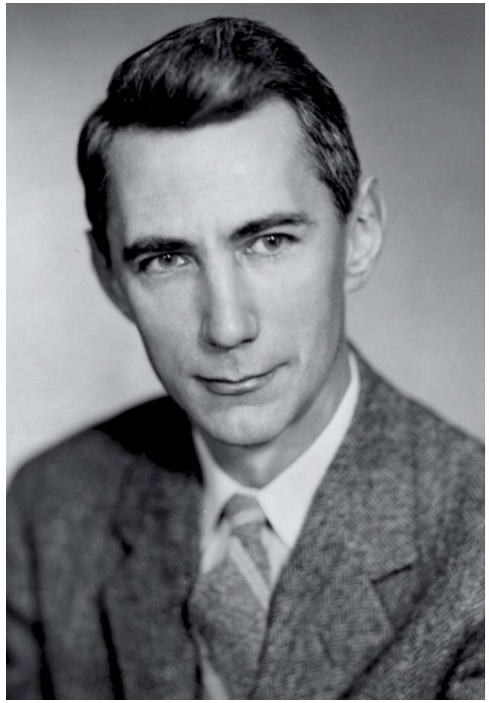
But: consider satellite TV with rain, N_0 gets bigger. So E_b/N_0 decreases. It depends on the noise too.

Shannon capacity* (1/3)

How can we evaluate the performance of a system?

$$R_b \leq \mathcal{C} = B \cdot \log_2\left(1 + \frac{S}{N}\right) = B \cdot \log_2\left(1 + \frac{E_b}{N_0} \cdot \frac{R_b}{B}\right)$$

Shannon capacity



Communication Theory of Secrecy Systems*

By C. E. SHANNON

1. Introduction and Summary

THE problems of cryptography and secrecy systems furnish an interesting application of communication theory.[1] In this paper a theory of secrecy systems is developed. The approach is on a theoretical level and is intended to complement the treatment found in standard works on cryptography.[2] There, a detailed study is made of the many standard types of codes and ciphers, and of the ways of breaking them. We will be more concerned with the general mathematical structure and properties of secrecy systems.

The treatment is limited in certain ways. First, there are three general types of secrecy system: (1) concealment systems, including such methods as invisible ink, concealing a message in an innocent text, or in a fake covering cryptogram, or other methods in which the existence of the message is concealed from the enemy; (2) privacy systems, for example speech inversion, in which special equipment is required to recover the message; (3) "true" secrecy systems where the meaning of the message is concealed by cipher, code, etc., although its existence is not hidden, and the enemy is assumed to have any special equipment necessary to intercept and record the transmitted signal. We consider only the third type—concealment systems are primarily a psychological problem, and privacy systems a technological one.

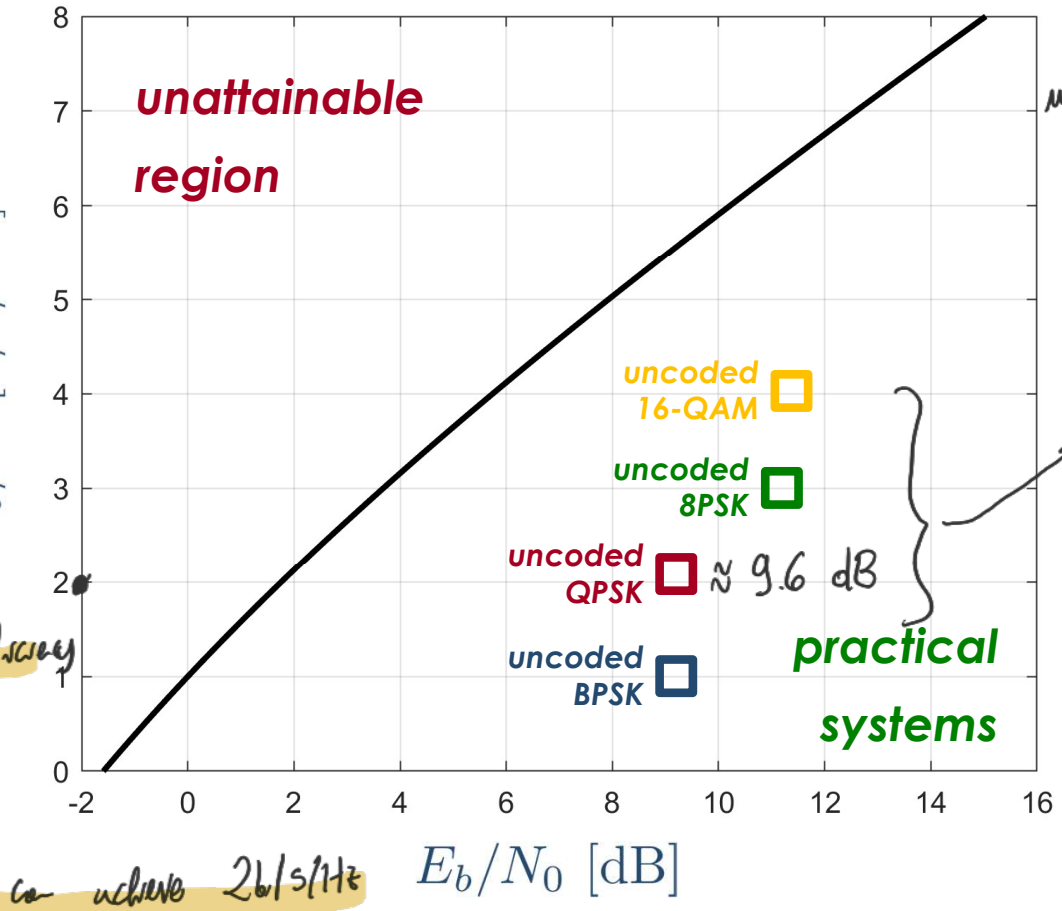
Secondly, the treatment is limited to the case of discrete information, where the message to be enciphered consists of a sequence of discrete symbols, each chosen from a finite set. These symbols may be letters in a language, words of a language, amplitude levels of a "quantized" speech or video signal, etc., but the main emphasis and thinking has been concerned with the case of letters.

The paper is divided into three parts. The main results will now be briefly summarized. The first part deals with the basic mathematical structure of secrecy systems. As in communication theory a language is considered to

* The material in this paper appeared originally in a confidential report "A Mathematical Theory of Cryptography" dated Sept. 1, 1945, which has now been declassified.
[1] Shannon, C. E., "A Mathematical Theory of Communication," *Bell System Technical Journal*, July 1948, p. 379; Oct. 1948, p. 623.
[2] See, for example, H. F. Gaines, "Elementary Cryptanalysis," or M. Givierge, "Cours de Cryptographie."

656

Shannon capacity* (2/3)



Divides the systems in practical systems and unattainable region.

This is without channel coding. We can get closer to the limit.

Ex:
I want $R_b = 20$ Mb/s
$B = 10$ MHz
$\frac{R_b}{B} = \frac{20\text{ Mb/s}}{10\text{ MHz}} = 2$ b/s/Hz

Let's say my Energy Efficiency is 10 dB. I'm in the practical systems region.

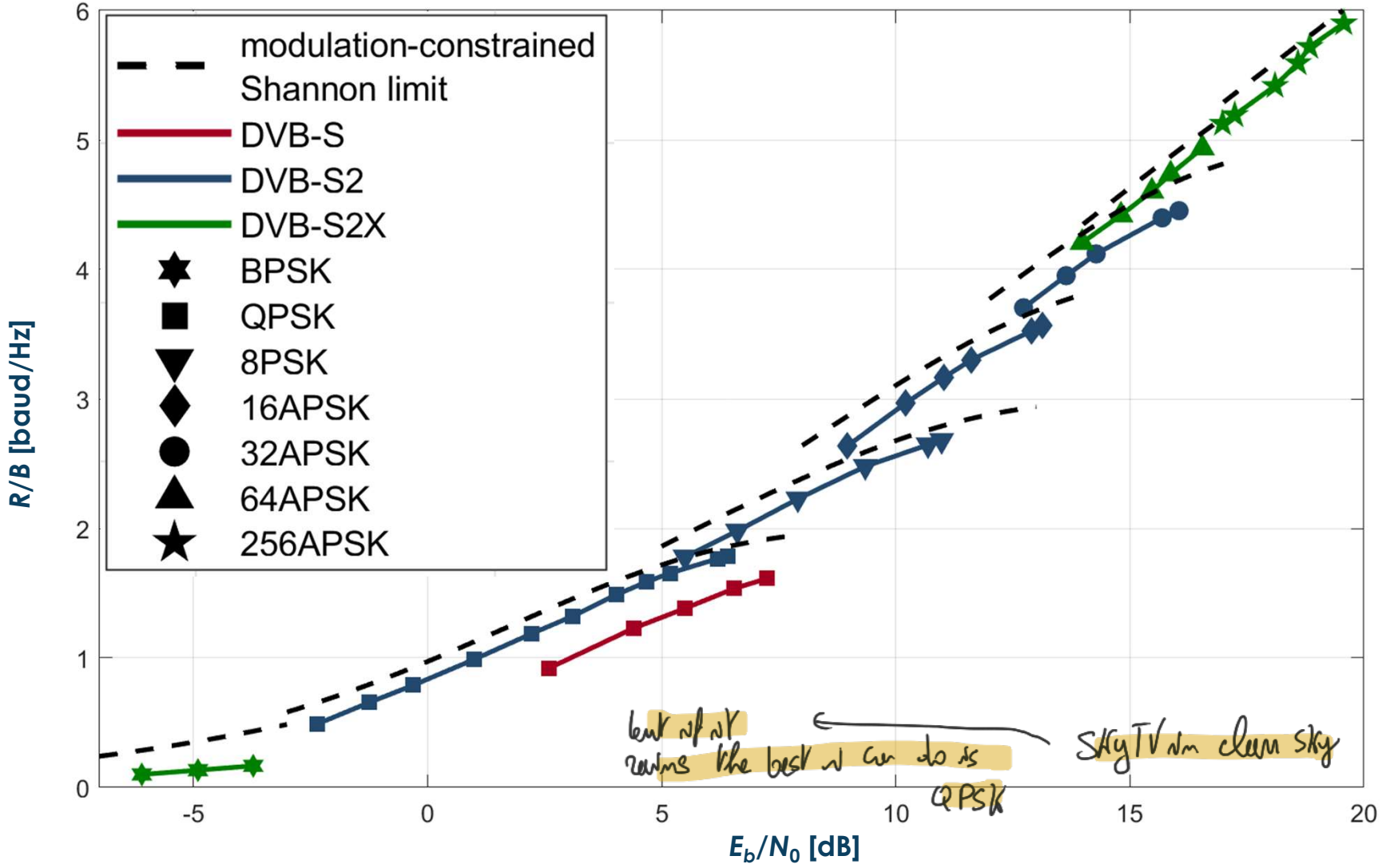
Shannon tells me I can achieve 2 b/s/Hz with a low BER. Otherwise, I can do it, but I will have a very high bit error rate. No matter how smart I am, there is no solution to this problem.

Dip. Ingegneria dell'Informazione
University of Pisa, Pisa, Italy
Giacomo Bacci

All the systems in the practical systems region can achieve an arbitrary low BER

Shannon capacity* (3/3)

Example: adaptive coding and modulation used by satellite standards



This is adaptive modulation and coding.

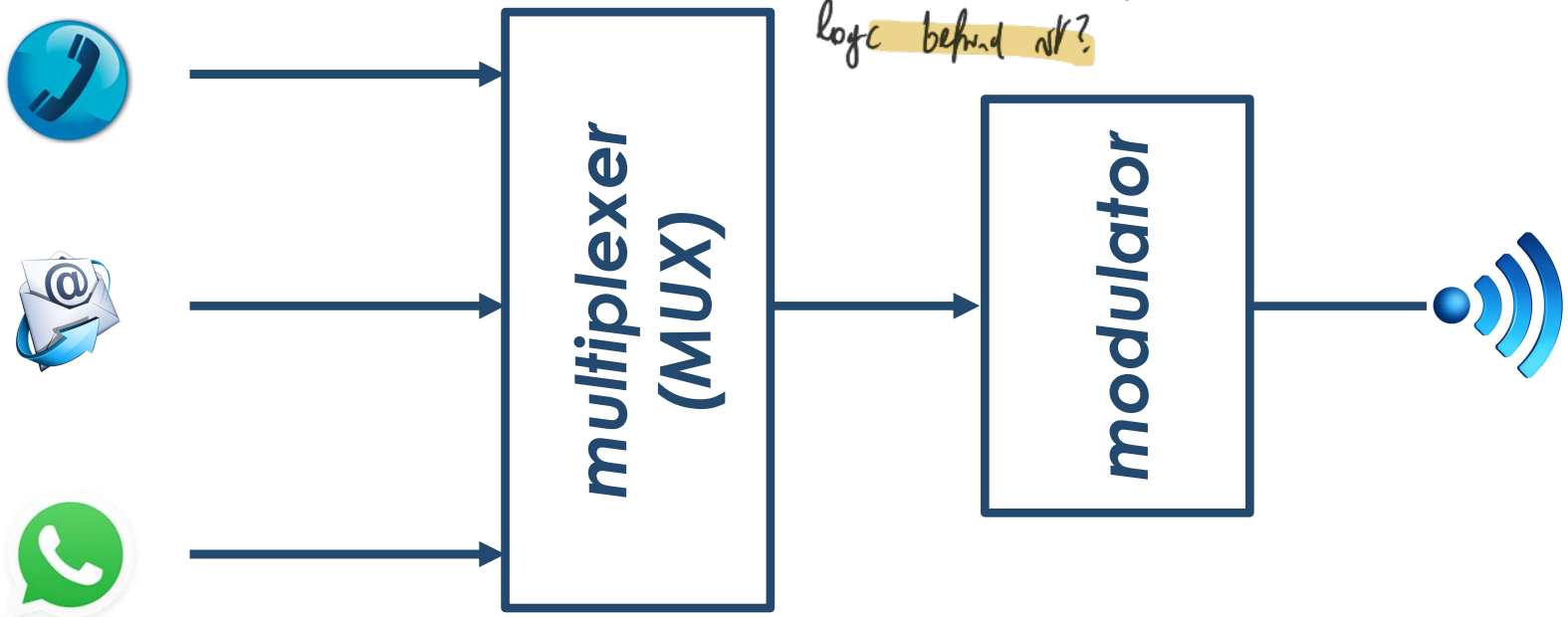
We have to introduce the energy efficiency to talk about optimal choices for our system.

Multiplexing

Multiplexing vs. multiple access (1/3)

Multiplexing: separating different flows at the same transmit side

Used to put different streams together. The modulator is blind to what happens before. How to arrange the bits to maintain the logic behind it?

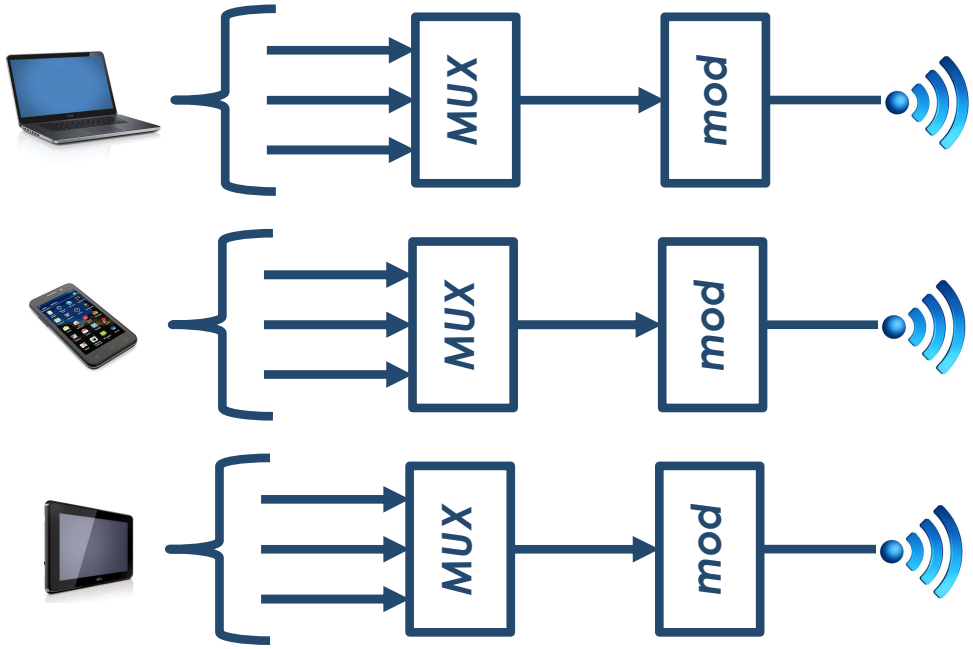


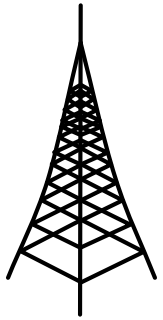
Diff. streams, same device

Multiplexing vs. multiple access (2/3)

Multiple access: separating different users at the receiver side

diff. users, same network, with multiple access schemes.



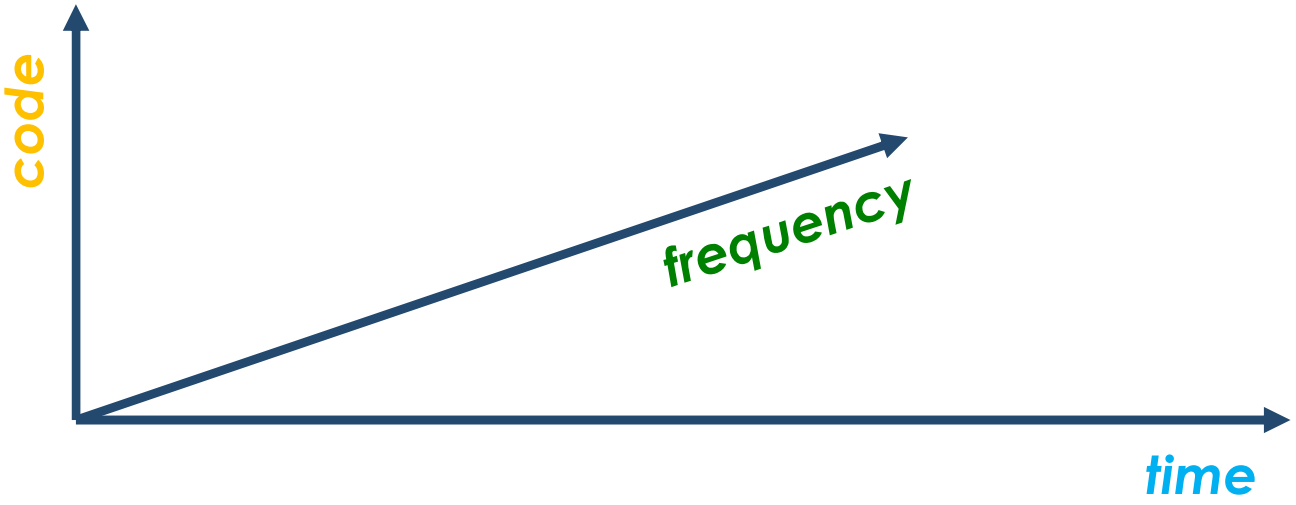


LEZIONE 9
END

Similar, but: imagine cellular network: if you generate streams from 1 device, you have coordination, no delay. In the second case you have different delays and propagation times. The solutions might be different. MA needs to deal with propagation delay, multiplexing doesn't.

Multiplexing vs. multiple access (3/3)

We can exploit several **degrees of freedom**: frequency, time, space, codes, etc.



Time division multiplexing (TDM)

Suppose you have 3 streams: on your phone you are sending an email and at the same time you are calling someone and sending a picture. 3 signals.

How to combine the 3 strings? Easiest approach:

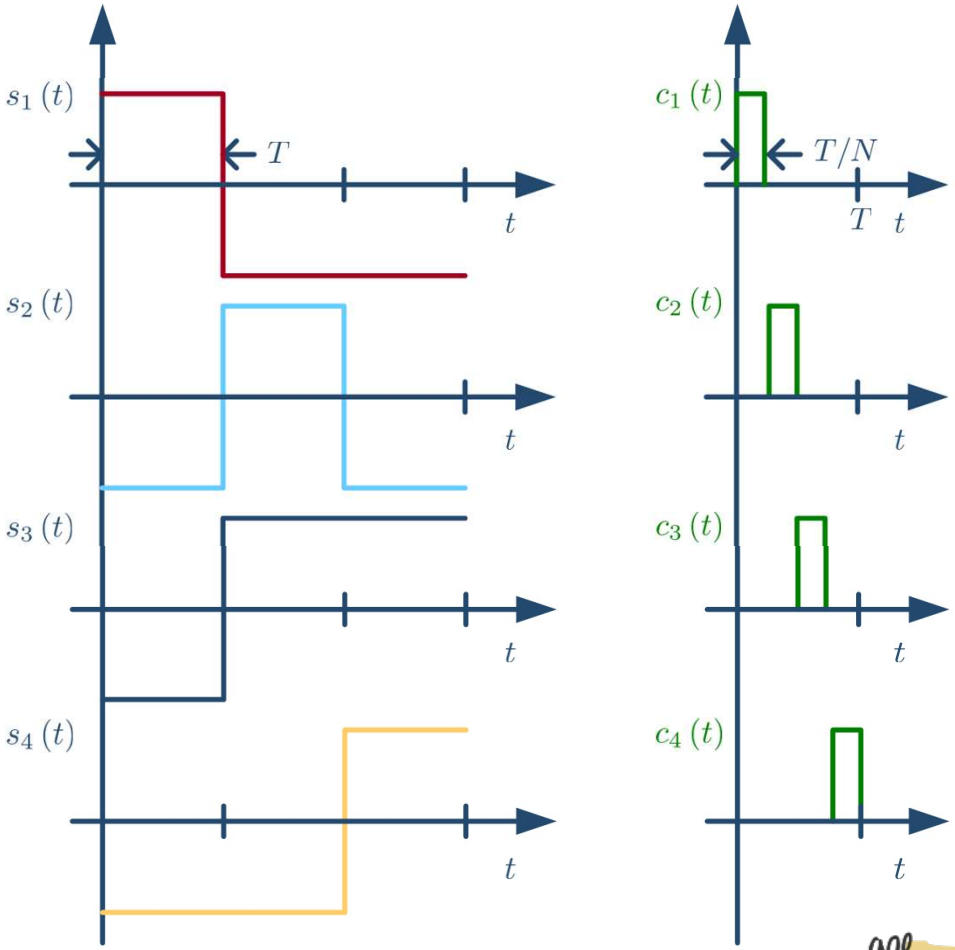
you take 1st bit of all 3 strings, then the 2nd ... etc.

And then you can treat it just like one stream! So modulation is normal.

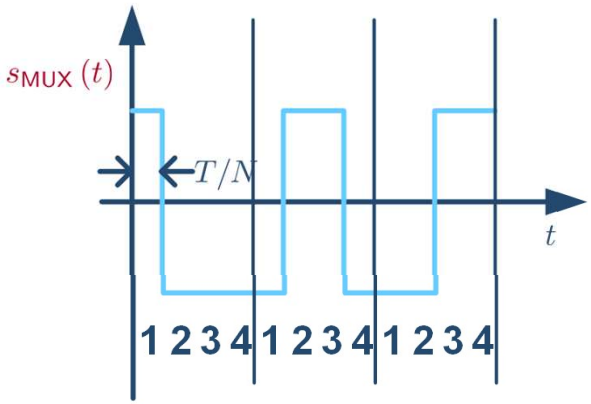
To do so, either you increase the bandwidth or reduce the bitrate

Time division multiplexing (TDM) (1/2)

In TDM, we can assume signature waveforms as NRZ binary signals at rate $R_c = N \cdot R = N/T$:



There is no overlapping between the different streams.



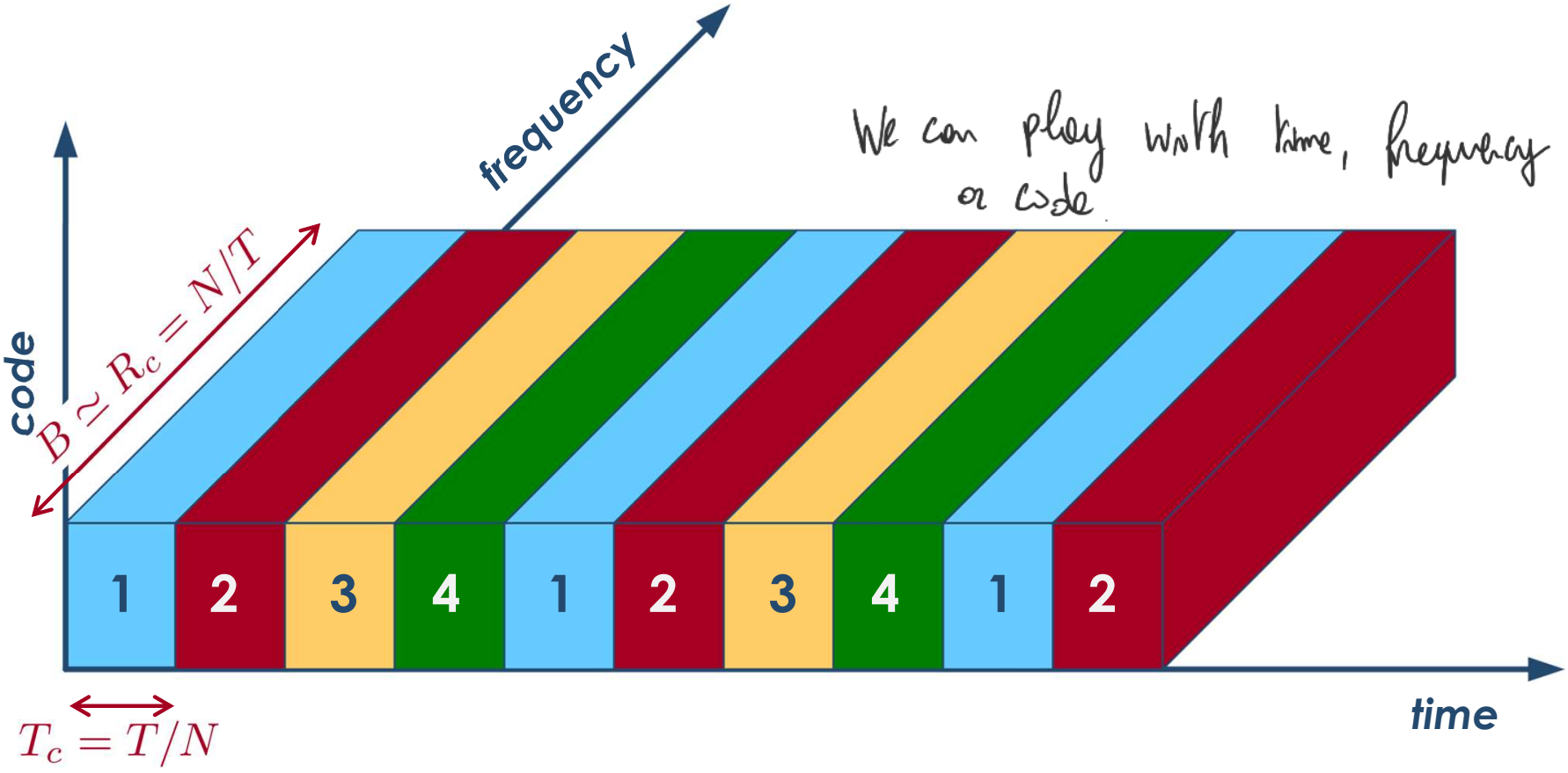
We start from a $\frac{1}{T}$ bit rate. In the end we reduce the bit interval to $\frac{T}{N}$, so the bit rate increases to $\frac{N}{T}$, so we are maintaining the same bit for each stream so we need an increased band.

The stream will be sent to the channel, then the demultiplexer is able to reconstruct the different streams.

Time division multiplexing (TDM) (2/2)

control plane tells you info for interpreting the signal and receive the data from data plane. Management plane is the third.

Each stream makes use of the whole bandwidth using a round robin scheduling, with time slot duration $T_c = T/N$:



We can play with time, frequency or code.

Playing in the frequency domain is usually done with analog signal

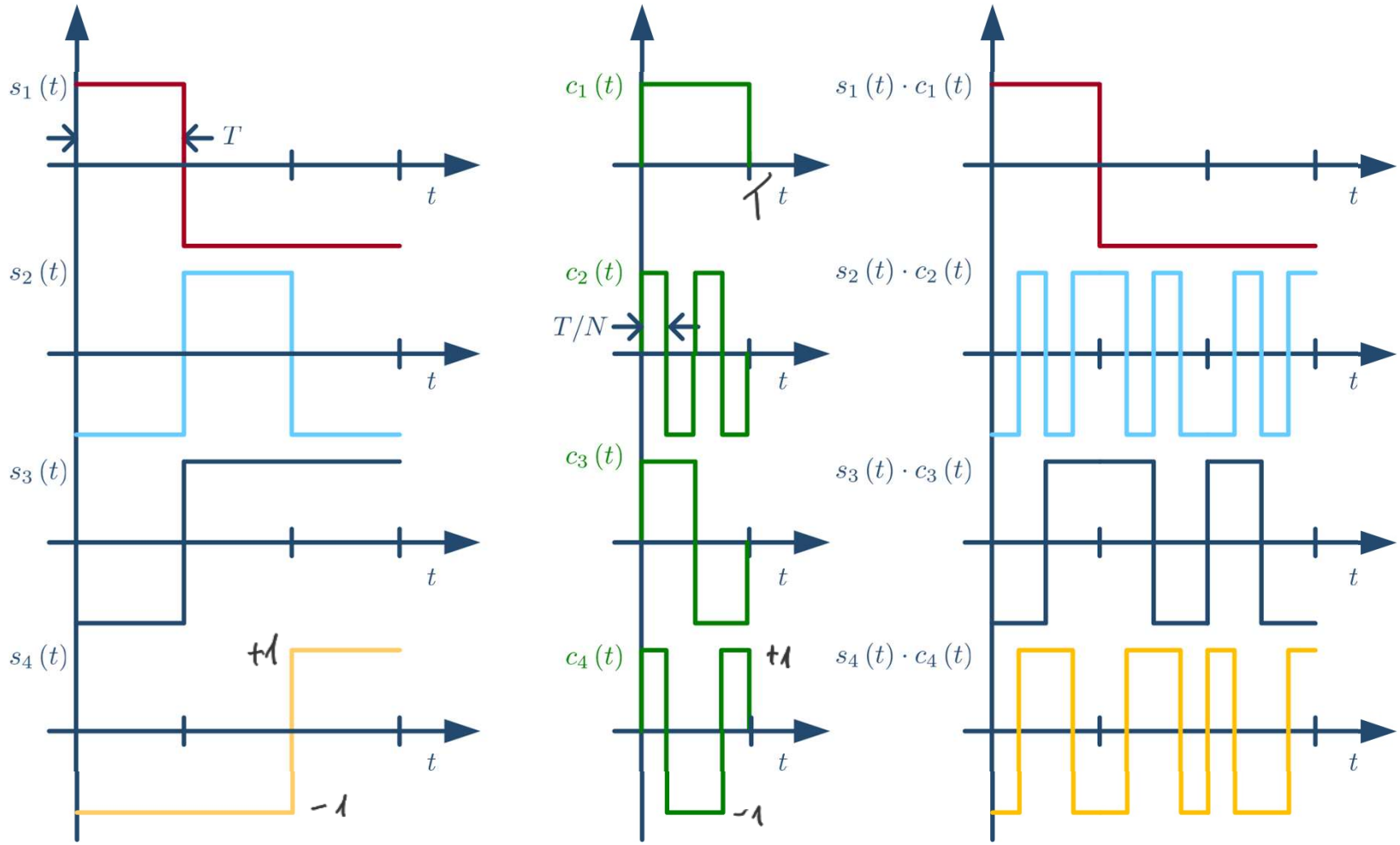
Practical applications of TDM*

- **E1 (Europe) & T1 (USA and Japan): byte-based TDM**
- **NRZ binary signals, with $R = 64\,\mathrm{kb/s}$ and $T = 15.625\,\mu\mathrm{s}$ are grouped byte by byte: $T_B = 8T = 125\,\mu\mathrm{s}$**
- **The E1 multiplex includes *N*=32 streams (30: data plane, 2: control plane)**
- **The TDM signal shows $R_c = N \cdot R = 2.048\,\mathrm{Mb/s}$ and $T_c = T/N \simeq 0.49\,\mu\mathrm{s}$ (for a re-clocked byte time $\simeq 3.9\,\mu\mathrm{s}$)**

Code division multiplexing (CDM)

Code division multiplexing (CDM) (1/4)

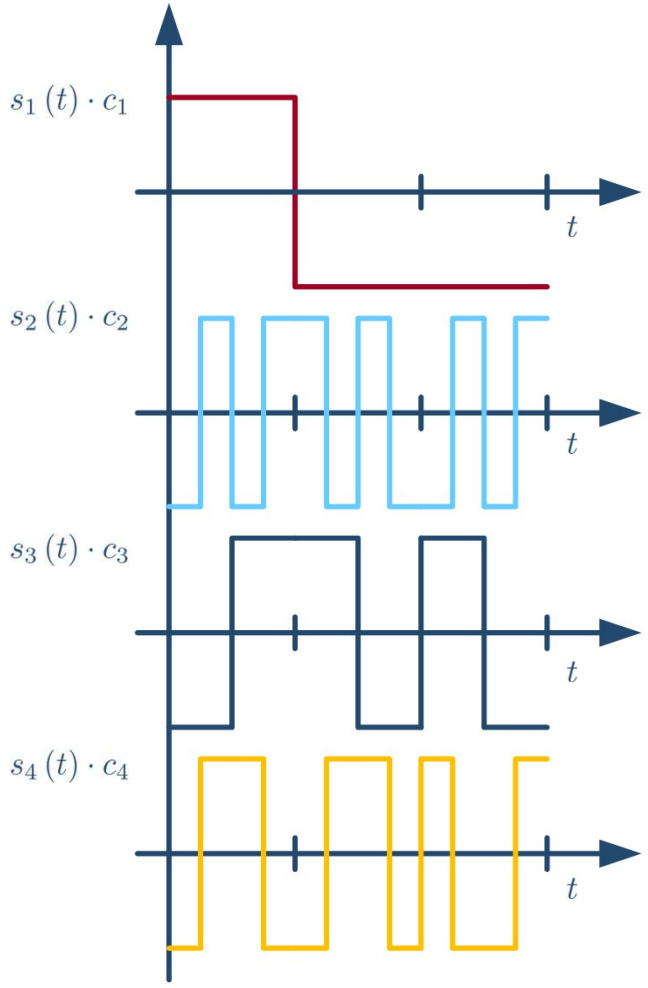
In CDM, signatures are given by a code set $\{c_n(t)\}_{n=1}^{N}$:

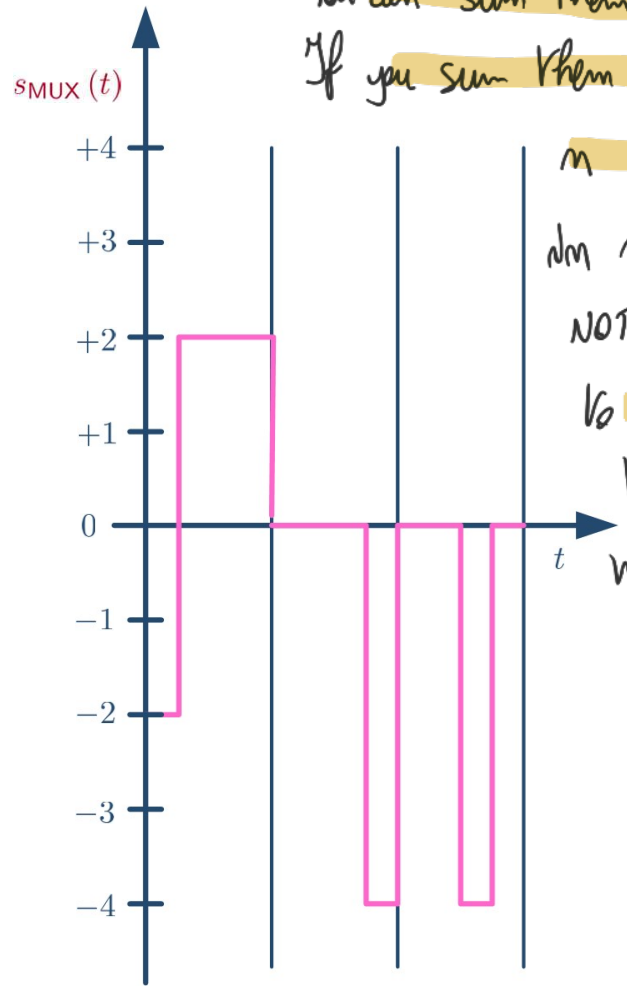


Here we have the same 4 users. We assign them different codes, not slots.
+1 = logical 1, -1 = logical 0. You multiply the signal by the code, which is an XNOR: 00 = 1, 11 = 1, 01 = 0, 10 = 0. Length of the code is the same as the time slot.

Code division multiplexing (CDM) (2/4)

The CDM-multiplexed signal is the summation of all coded streams:





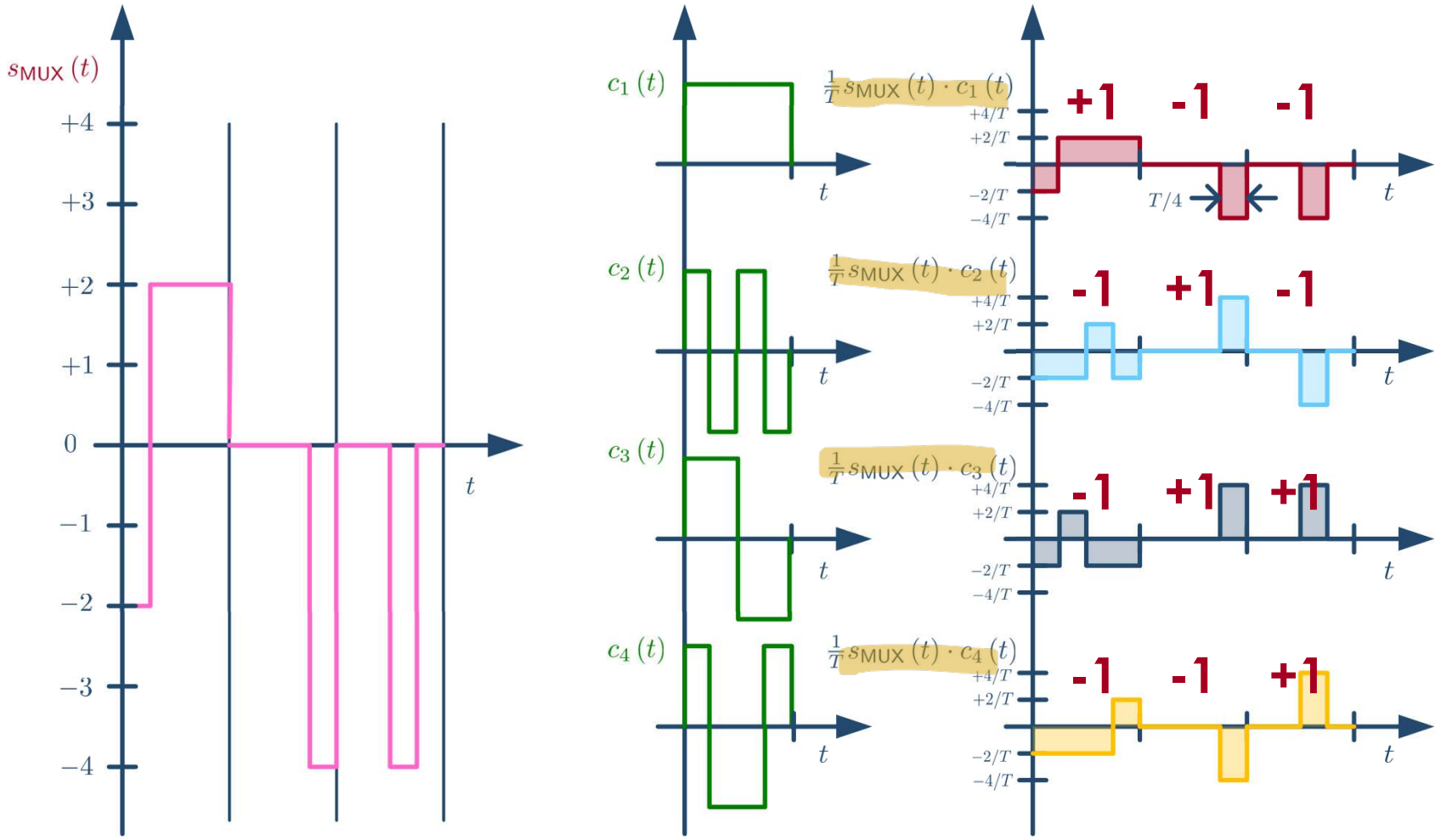
You can sum them in the natural domain.
If you sum them all together you have n streams, resulting in n+1 possible values.
NOTE: this line needs to be squeezed to use the same power you would normally use.

Dip. Ingegneria dell'Informazione
University of Pisa, Pisa, Italy

So here we have 4 different streams always overlapping. Can they be decoded?

Code division multiplexing (CDM) (3/4)

To decode the *n*th stream, the receiver needs to know the code set:



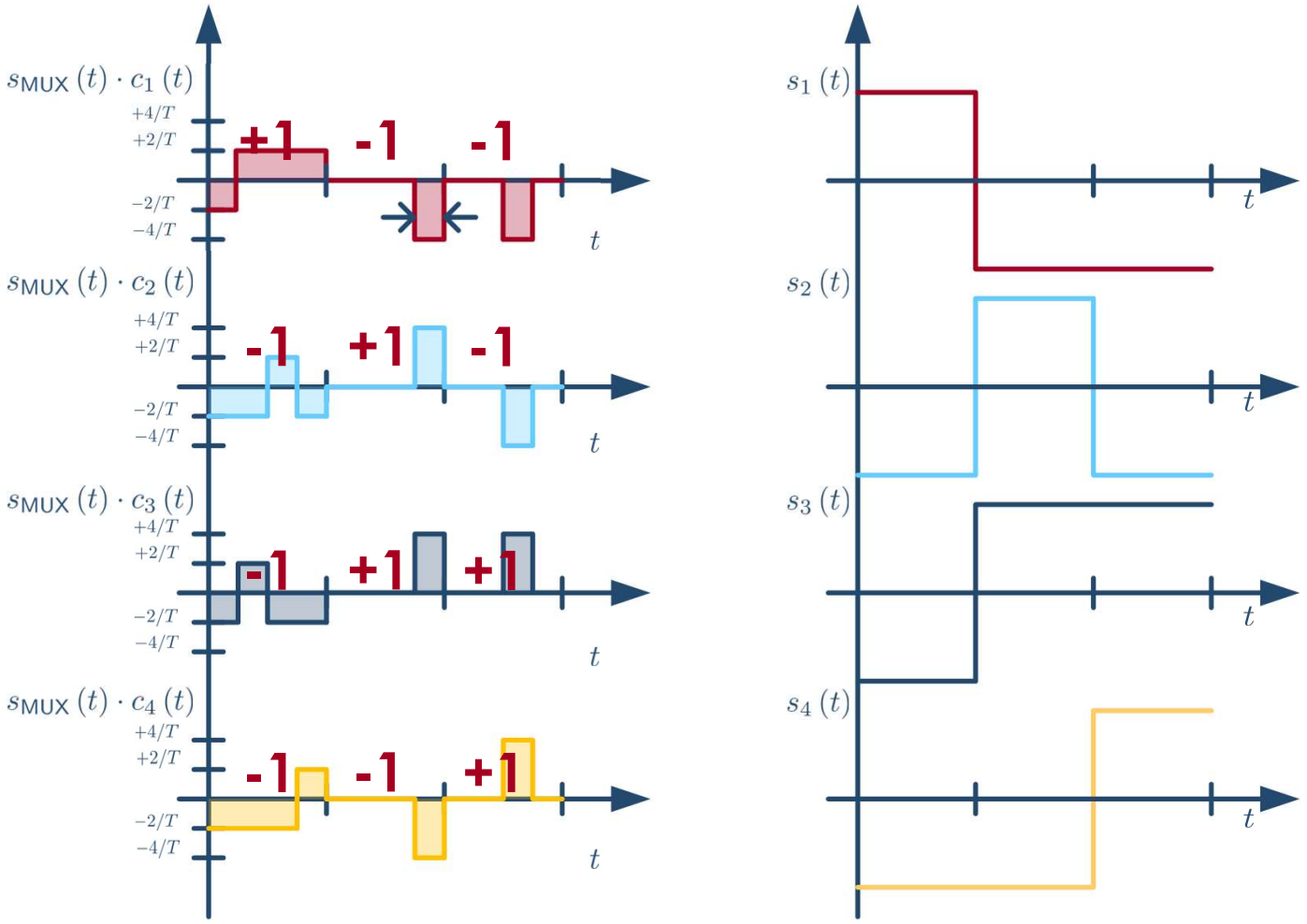
Demultiplexer uses the exact same code used by the multiplexer. If you multiply the whole curve by the codes, you are able to decode. The demultiplexer knows when streams start and end. You need to know when each stream starts and ends. You need synchronization.

After the multiplication, how do we interpret all of that? We need to integrate over the bit interval. The area of each rectangle is (for yellow): $-\frac{3}{4}T \cdot \frac{2}{T} = -\frac{3}{2}$

$+\frac{1}{4}T \cdot \frac{2}{T} = \frac{1}{2} \Rightarrow -1 = 0$

Code division multiplexing (CDM) (4/4)

To decode the *n*th stream, the receiver needs to know the code set:



Walsh-Hadamard codes (1/4)

A useful set of signature codes for CDM is the Walsh-Hadamard (WH) code set

The *n*th code of the WH set is represented by the *n*th row of the *N*-order Hadamard matrix $\mathbf{H}_N$:

$$H_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

4 submatrixes, where the 4th one is H_2 complemented (when 0, 1, when 1, 0)

$$\mathbf{H}_4 = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix} \begin{matrix} c_1(t) \\ c_2(t) \\ c_3(t) \\ c_4(t) \end{matrix}$$

$$H_2 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} H_1 & H_1 \\ H_1 & \overline{H_1} \end{pmatrix}$$

Walsh-Hadamard codes (2/4)

The *N*-order WH code set, with cardinality $N = 2^W$, with $W \in \mathbb{N}$, can be obtained in a recursive fashion:

$$\mathbf{H}_2 \triangleq \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix}, \quad \mathbf{H}_N = \mathbf{H}_2 \otimes \mathbf{H}_{N/2} = \begin{bmatrix} \mathbf{H}_{N/2} & \mathbf{H}_{N/2} \\ \mathbf{H}_{N/2} & \overline{\mathbf{H}}_{N/2} \end{bmatrix}$$

where $\otimes$ is the Kronecker product, and $\overline{\mathbf{H}}$ is the modulo-2 complement of the matrix $\mathbf{H}$

Walsh-Hadamard codes (3/4)

Examples:

$$\mathbf{H}_2 \triangleq \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix}$$

$$\mathbf{H}_4 = \begin{bmatrix} \mathbf{H}_2 & \mathbf{H}_2 \\ \mathbf{H}_2 & \overline{\mathbf{H}}_2 \end{bmatrix} = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$$

$$\mathbf{H}_8 = \begin{bmatrix} \mathbf{H}_4 & \mathbf{H}_4 \\ \mathbf{H}_4 & \overline{\mathbf{H}}_4 \end{bmatrix} = \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 & +1 & -1 & -1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$$

Walsh-Hadamard codes (4/4)

Exercise: Show that this signature set $\{c_n(t)\}_{n=1}^{N}$ is orthogonal

Kronecker delta, if $n=m$, then 1 otherwise 0

$$\frac{1}{T}\int_0^T c_n(t)\cdot c_m^*(t)\,dt = \delta[n-m] \qquad \forall n,m$$

Thanks to this property, codes like the WH ones are called **orthogonal codes**

CDM vs. TDM (1/2)

Similarly to TDM, in CDM all streams need to be synchronized (hence, CDM is not suitable for analog systems)

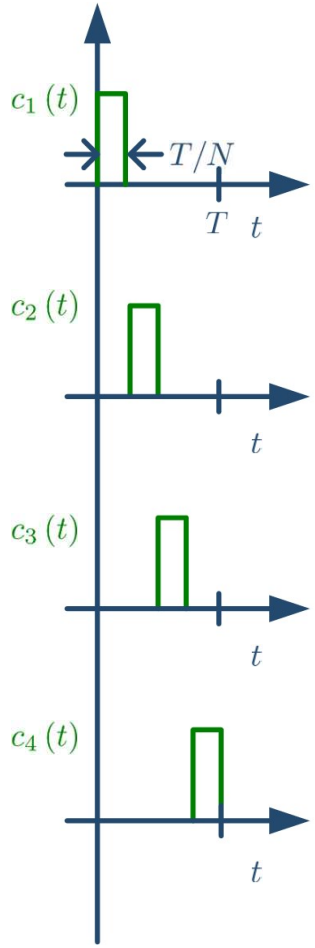
However, signatures in TDM are non-null only for a time slot, whereas codes in CDM are pseudo-random noise sequences:

TDM: $$\frac{1}{T}\int_0^T c_n(t)\,dt = \frac{1}{N}$$

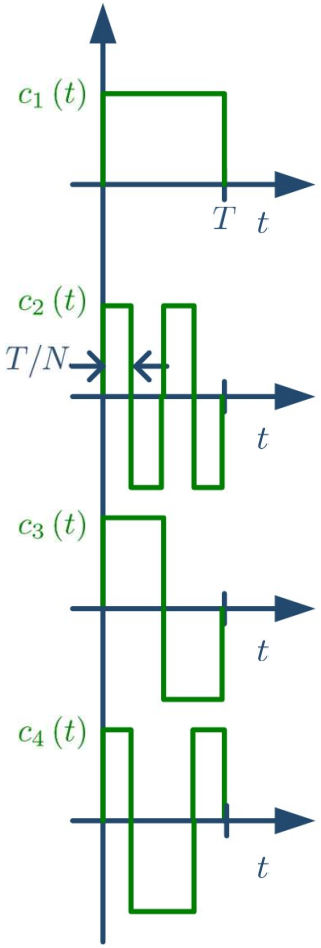
CDM: $$\frac{1}{T}\int_0^T c_n(t)\,dt = 0$$

CDM vs. TDM (2/2)

TDM: $\frac{1}{T}\int_0^T c_n(t)\,dt = \frac{1}{N}$

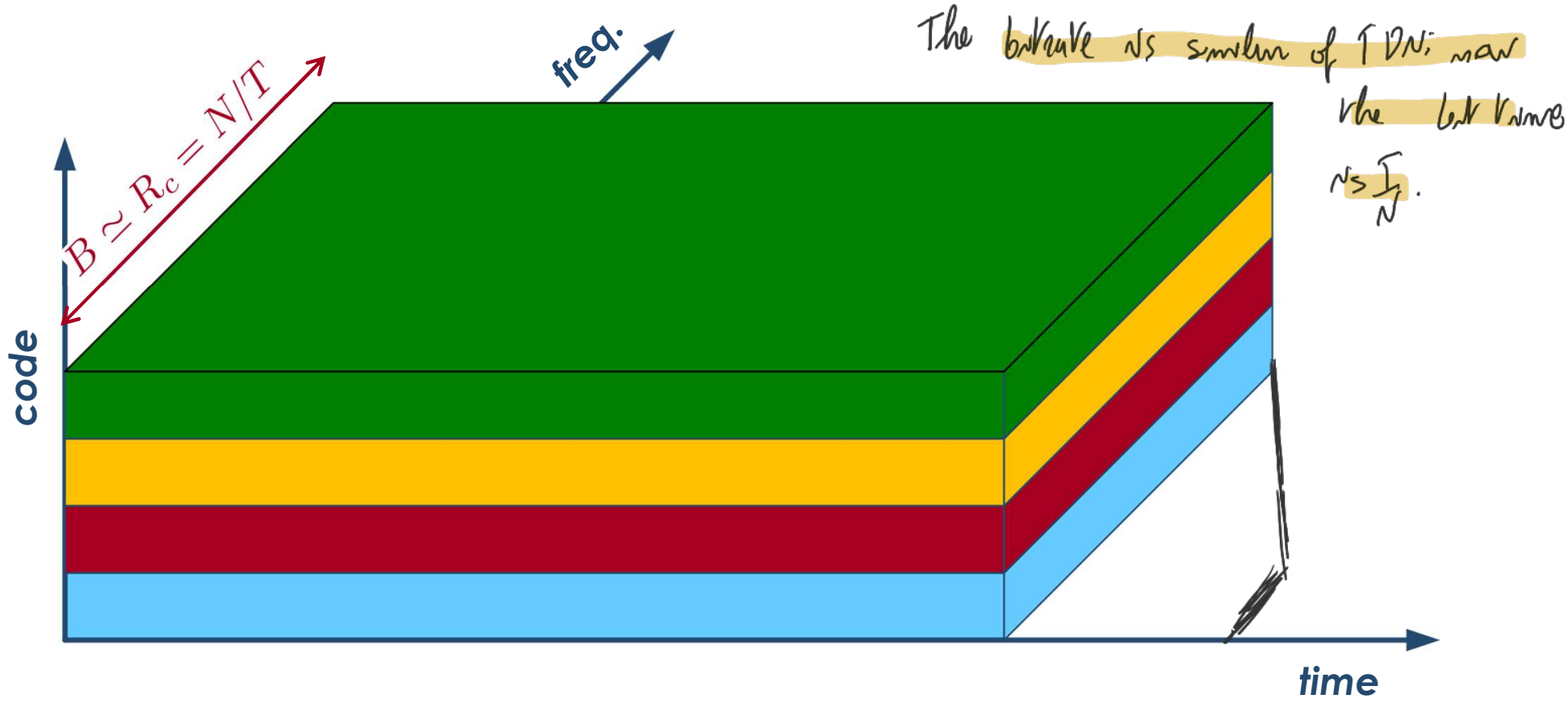


CDM: $\frac{1}{T}\int_0^T c_n(t)\,dt = 0$



Code division multiplexing (CDM)

This holds true for each CDM signal obtained with orthogonal codes: it makes continuous use of the whole bandwidth :



Multiple access techniques

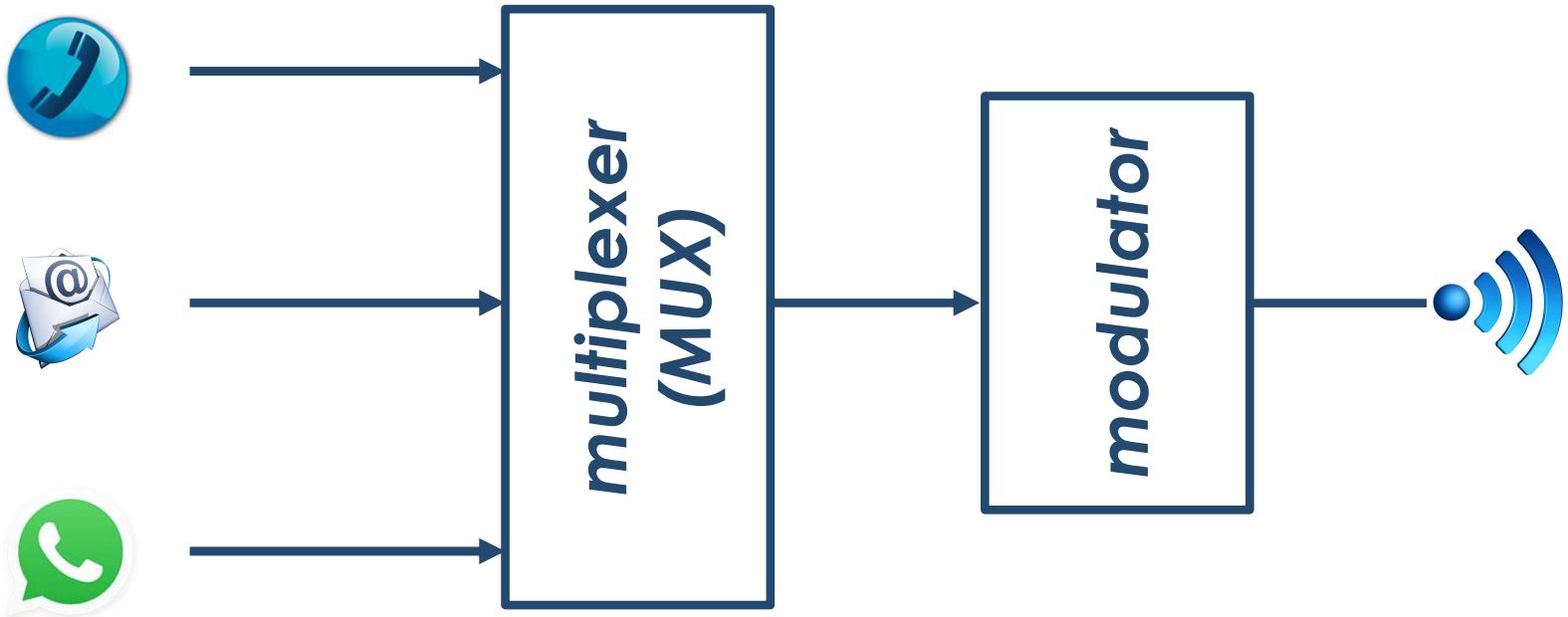
Multiplexing for different streams at the same transmitter.
MA different transmitters over the same system.

Biggest difference: the transmitters are separated. So the propagation can have different delays: receiver gets bits/signals at different times (microseconds, but we will still lose synchronization). We have to compare the bitrate and how much the time difference can be significative.

So the approach vs TDMA ☹ JMP ☺

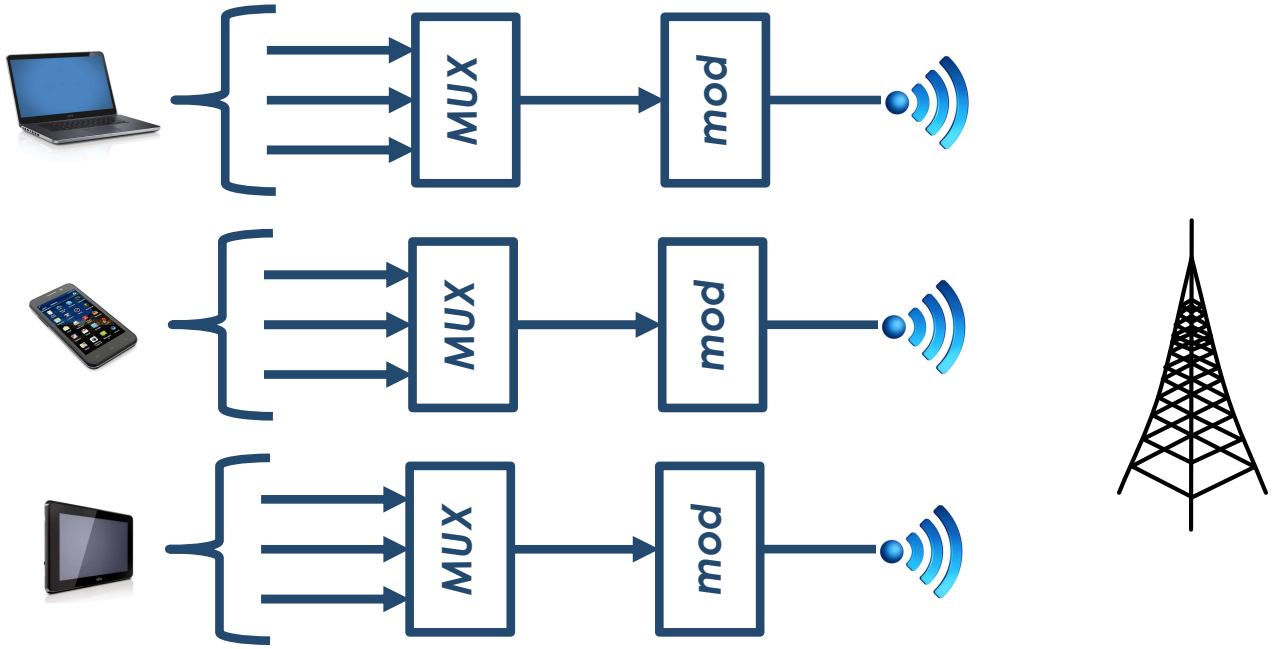
Multiplexing vs. multiple access (1/2)

Multiplexing: separating different flows at the same transmit side



Multiplexing vs. multiple access (2/2)

Multiple access: separating different users at the receiver side



Time division multiple access (TDMA)

JMP SLIDE 110

Time division multiple access (TDMA) (1/3)

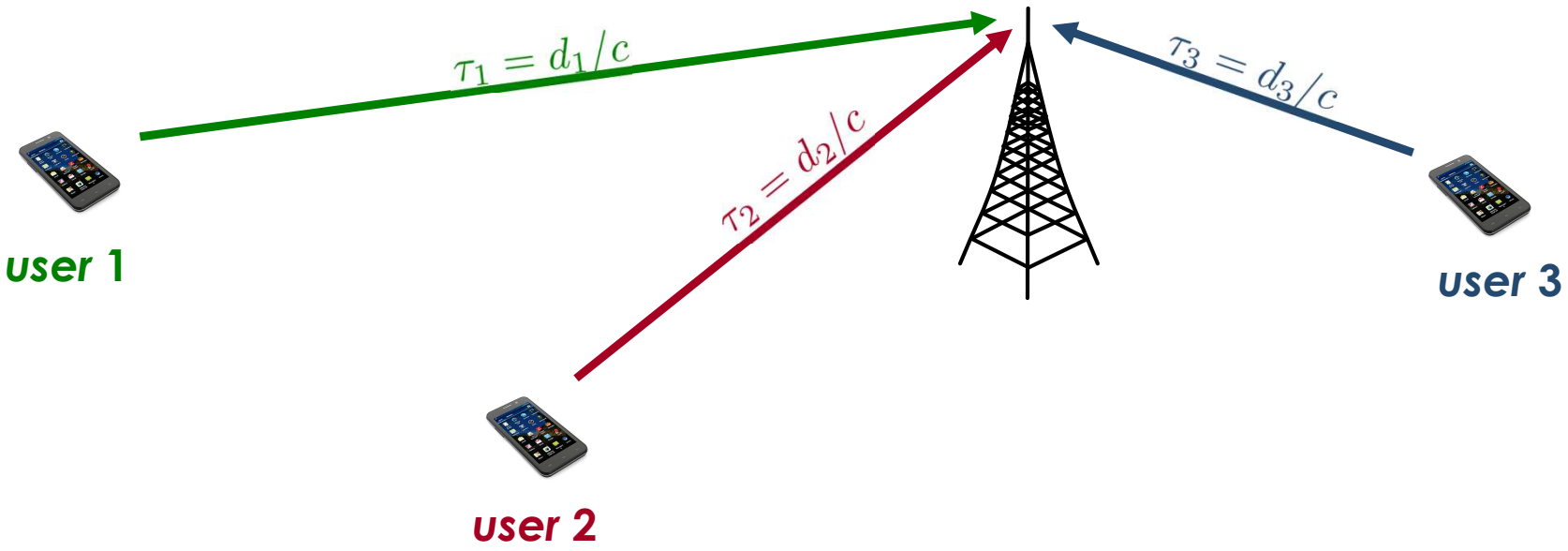
Unlike FDMA, TDMA presents some implementation issues concerning **time synchronization** across users (channels)

In addition to the channel allocation (similar to the FDMA), there is the need for **network synchronization** to align the users, which can be geographically sparse (and hence with different **propagation delays**)

Time division multiple access (TDMA) (2/3)

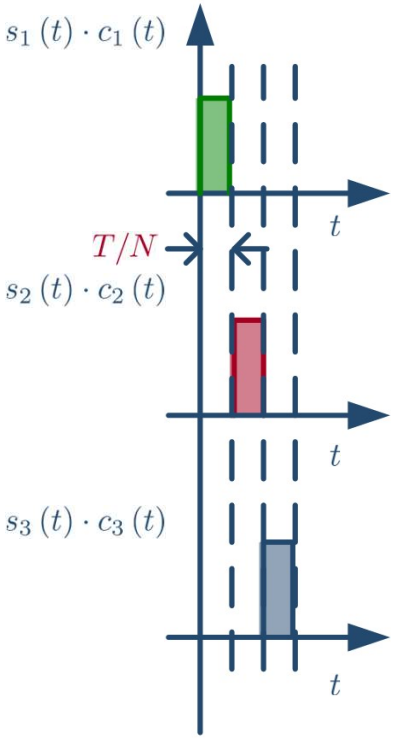
Example: a typical cellular network

Different users experience different propagation times.

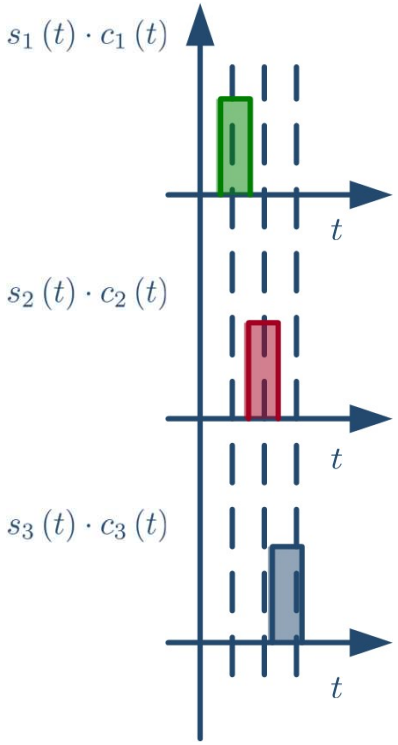


Time division multiple access (TDMA) (3/3)

Desired situation:



Actual situation:

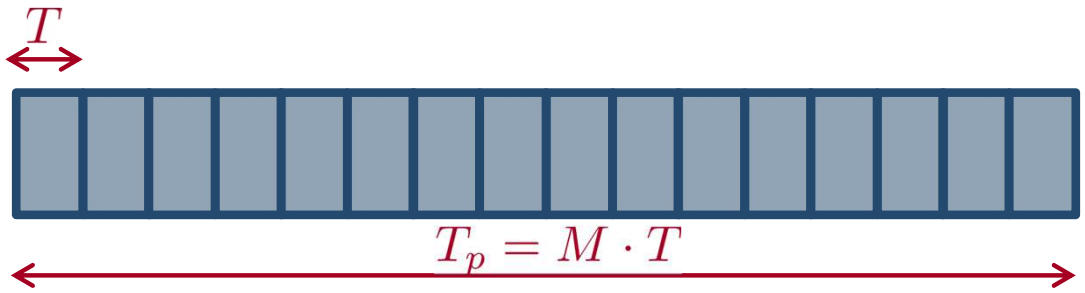


signals from different users overlap due to different propagation delays

Burst-based TDMA (1/2)

To simplify the synchronization tasks, time slots are grouped per burst instead of bit by bit

Each data burst is composed by a packet with duration $T_p = M \cdot T$, where M is the number of bits per packet

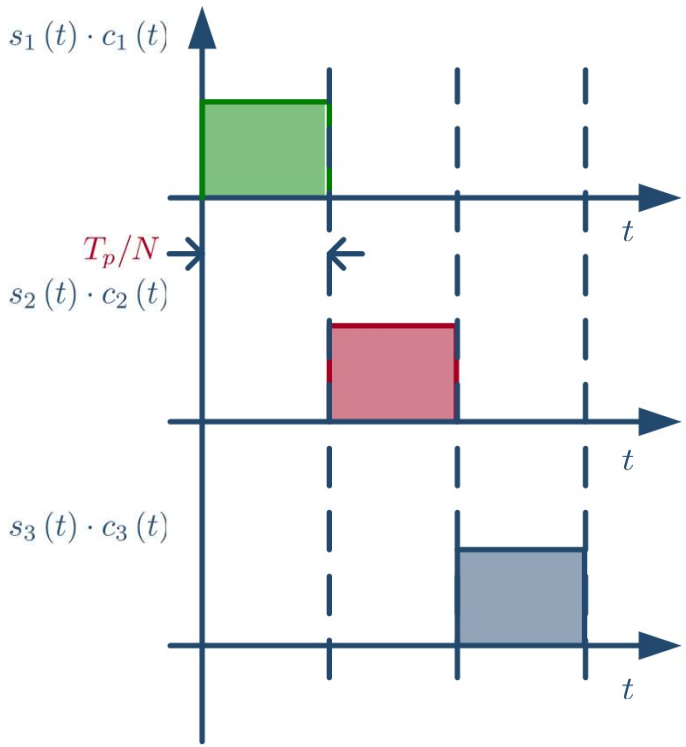


This approach allows the propagation times to be compared with T_p/N rather than with T/N (and hence there is a factor M)

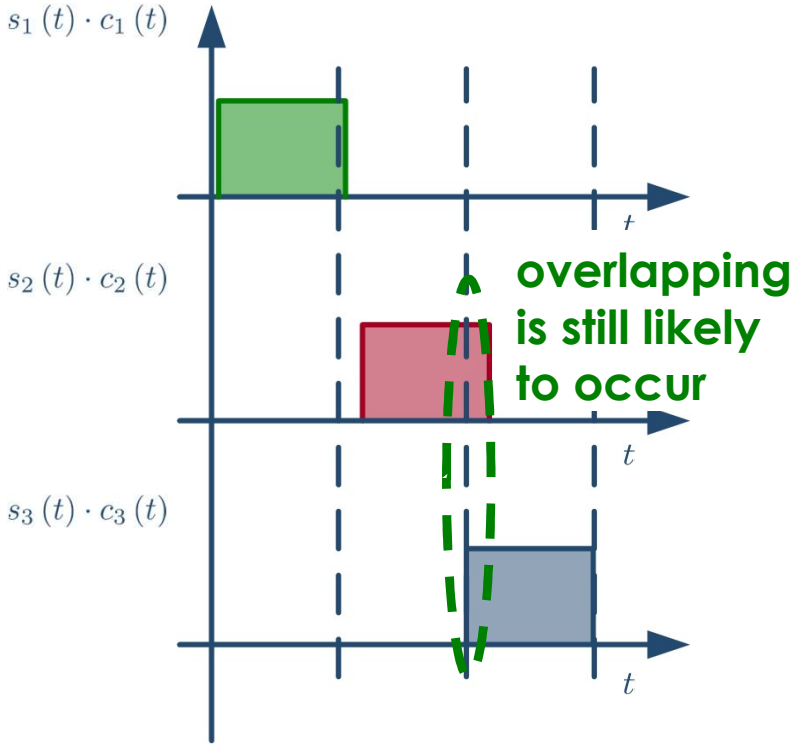
Burst-based TDMA (2/2)

The (centralized) network synchronization feeds back timing information to let the users anticipate or postpone transmissions

Desired situation:

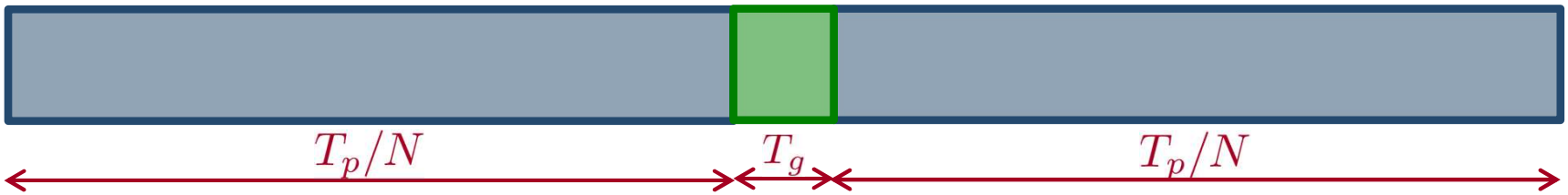


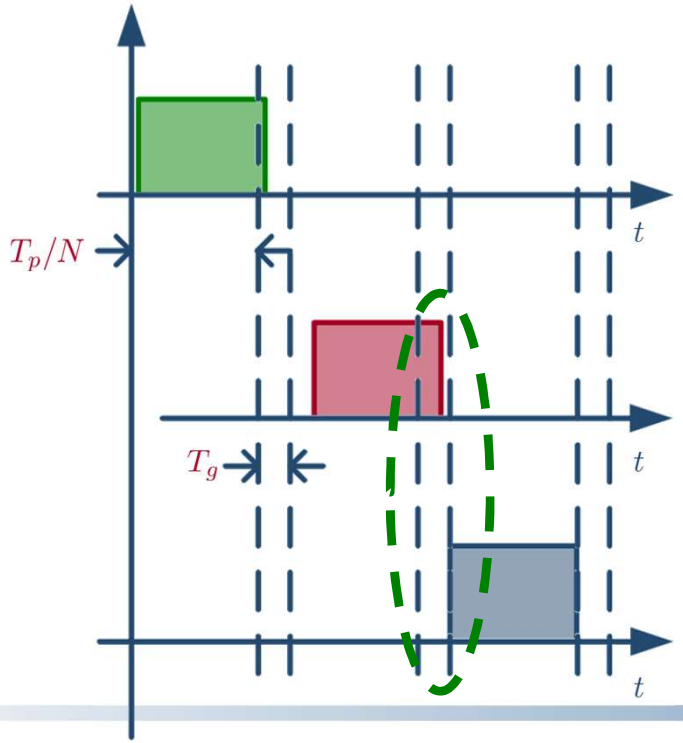
Actual situation:



Introducing a guard interval

To relax the network synchronization requirements, we can introduce a **guard interval** T_g between each burst

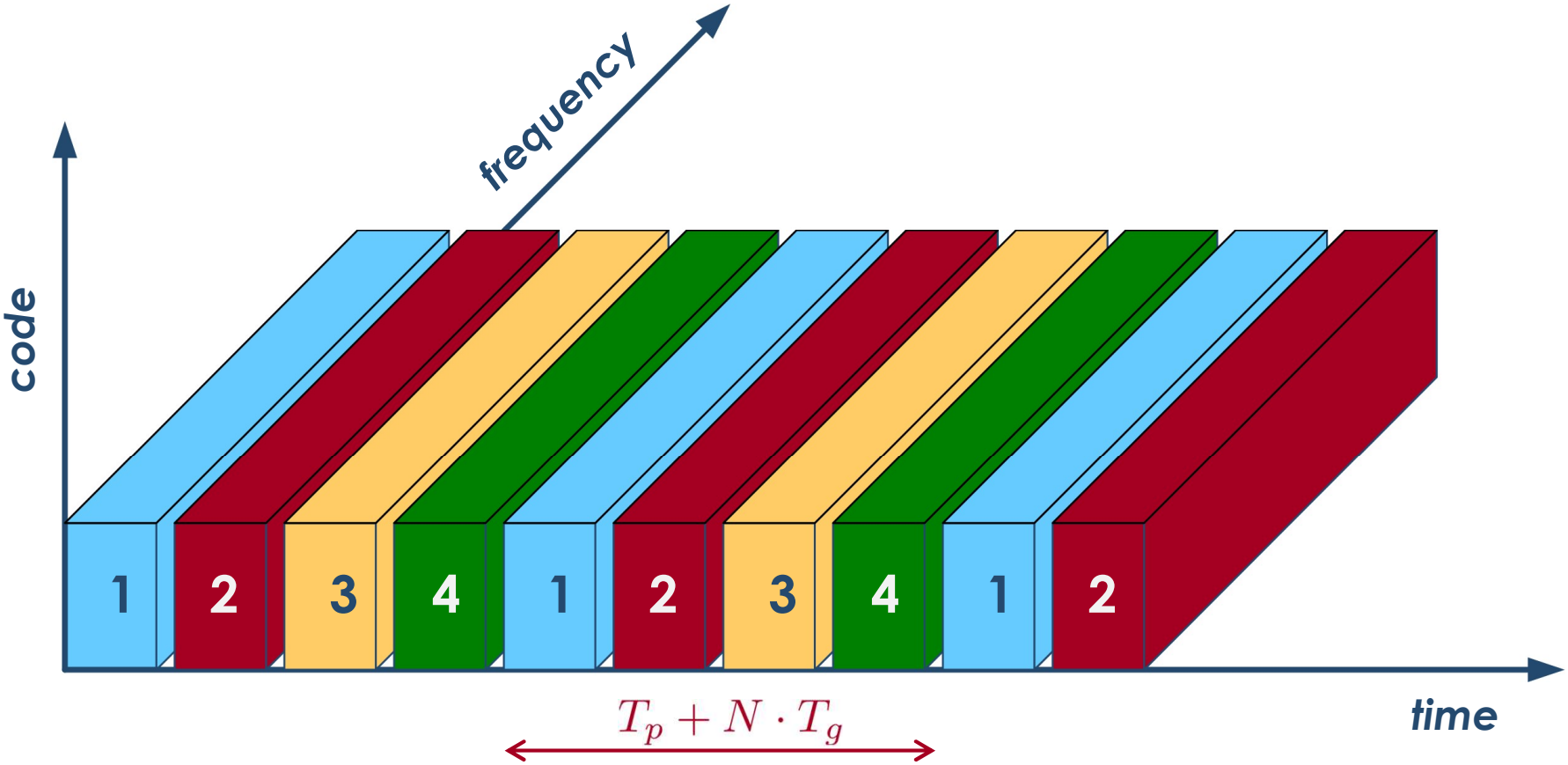




overlapping can more easily be avoided

TDMA in 3D resource plan

The guard interval is a tradeoff between synchronization performance and optimal usage of network resources



We want to separate the users in time domain. We need 3 modifications: 1. Instead of sending just 1 bit, larger packets of bits, so the delay because of the people talking (the difference in propagation time) is comparable to the delays. So we use slots of large bits

Ex: in my slot, you can talk, then in my slot I can talk.

1. Modification is extending the time slot.

2nd Modification: knowing the distance you start transmitting before to compensate the distance delay.

3rd. modification: guard interval in which no-one is allowed to transmit. In case of errors of possible overlapping there's still the gap created to protect.

RETURN ⊙

LEZIONE 10 END

Frequency division multiple access (FDMA)

Connected with modulation: the carrier frequency depends on MA and can vary for different users.

Frequency division multiple access (FDMA) (1/3)

Historically, the **first** multiple access technique adopted in radio communications is based on FDMA

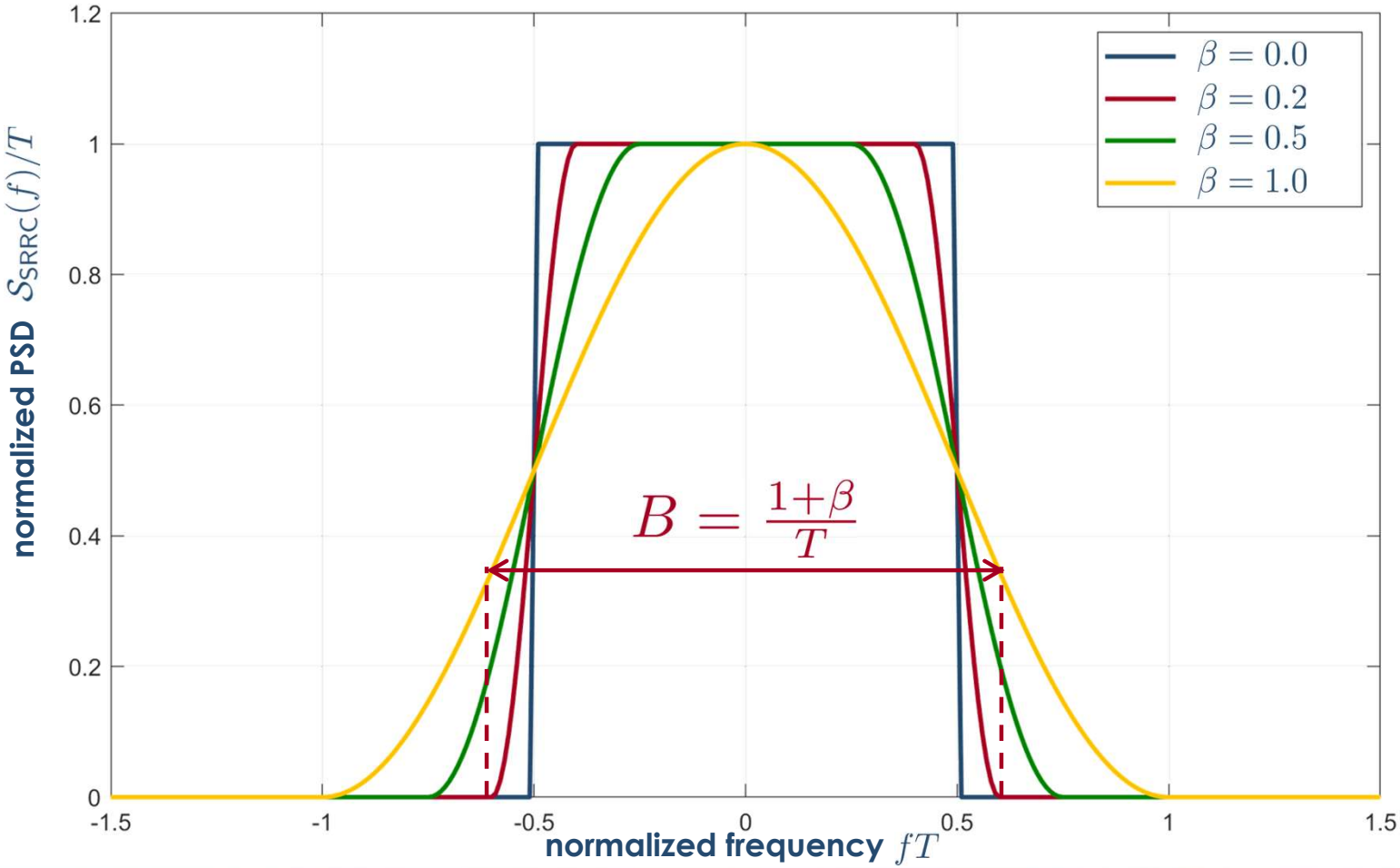
Typical application: analog amplitude-modulation (AM) radio channels

Constraint: each signal must be strictly **band-limited**

Note that there are **no** requirements in terms of time or phase alignment across different channels

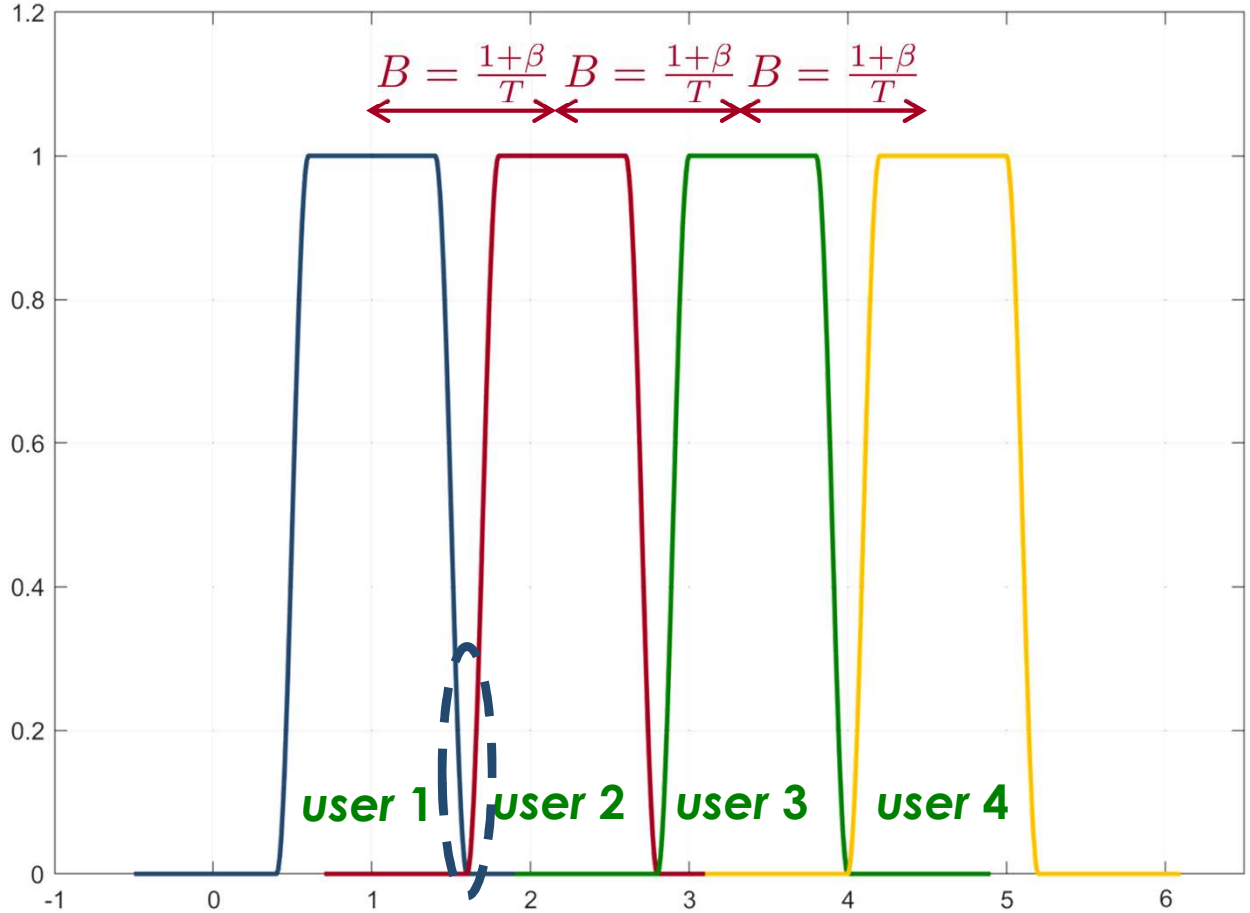
Frequency division multiple access (FDMA) (2/3)

Let us assume to consider a QAM signal modulated with a **square root raised cosine** (SRRC) shaping pulse:



Frequency division multiple access (FDMA) (3/3)

In the frequency domain, the multiple-access signal takes the form $\mathcal{S}_{\text{MA}}(f) = \sum_{n=1}^{N} \mathcal{S}_n\left(f - n \cdot \frac{1+\beta}{T}\right)$



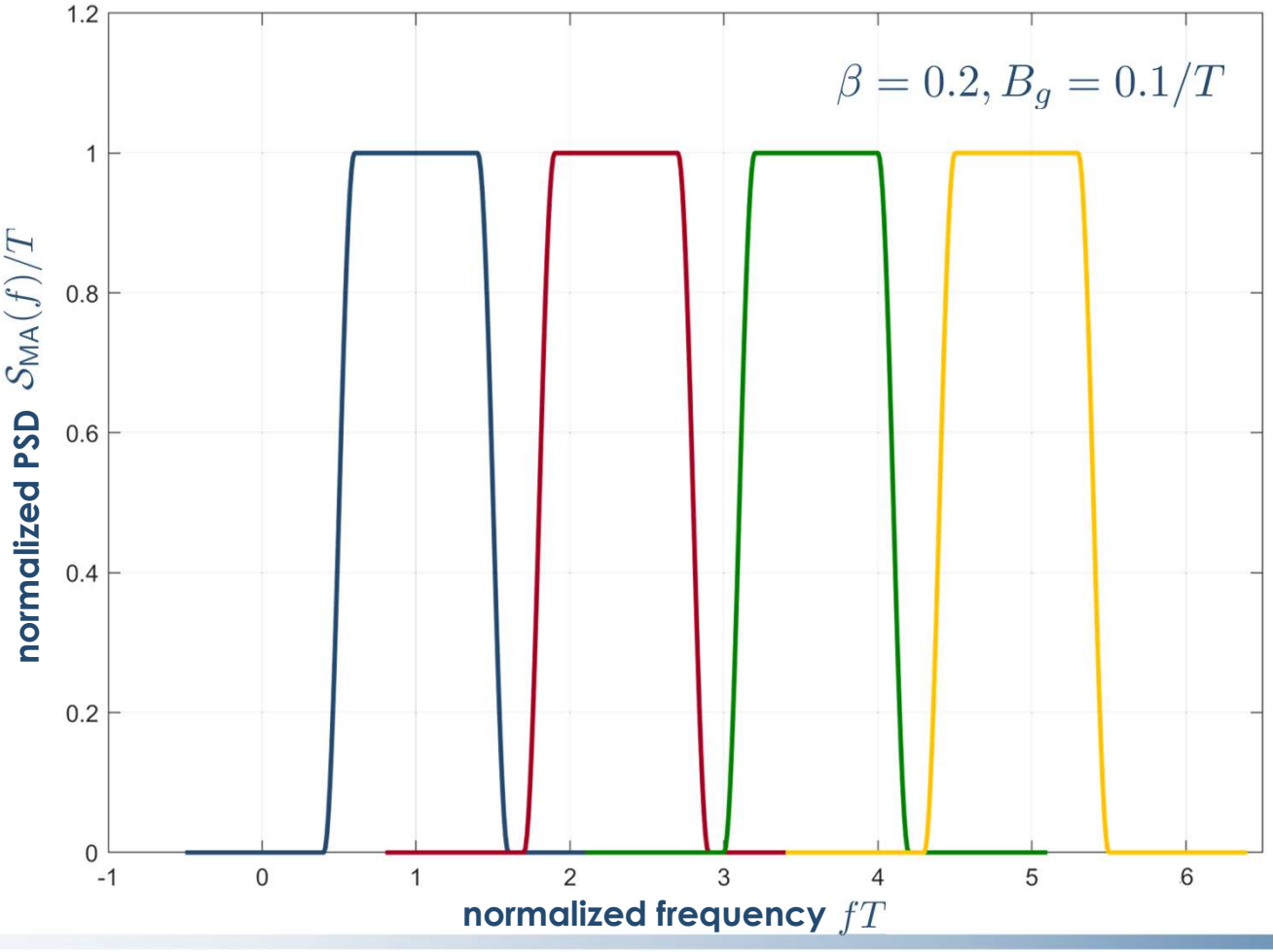
Introducing a guard bandwidth* (1/2)

To relax the requirements in terms of carrier frequency instabilities, we can add a guard bandwidth B_g to further separate the channel spectra:

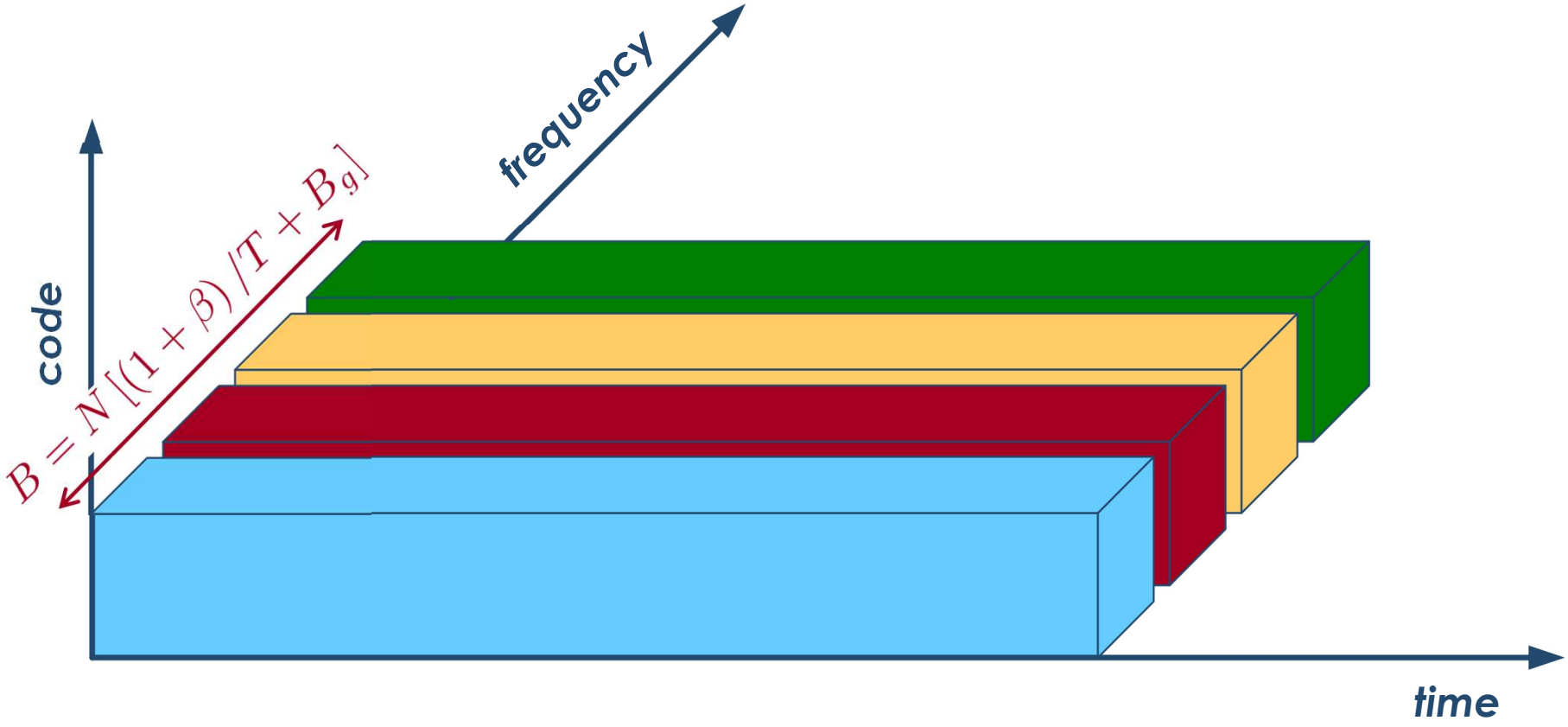
$$\mathcal{S}_{\text{MA}}(f) = \sum_{n=1}^{N} \mathcal{S}_n \left(f - n \left[\cdot \frac{1+\beta}{T} + B_g \right] \right)$$

Introducing a guard bandwidth (2/2)

Increased separation (and hence easier architecture at both tx and rx sides) comes at the cost of additional resource waste

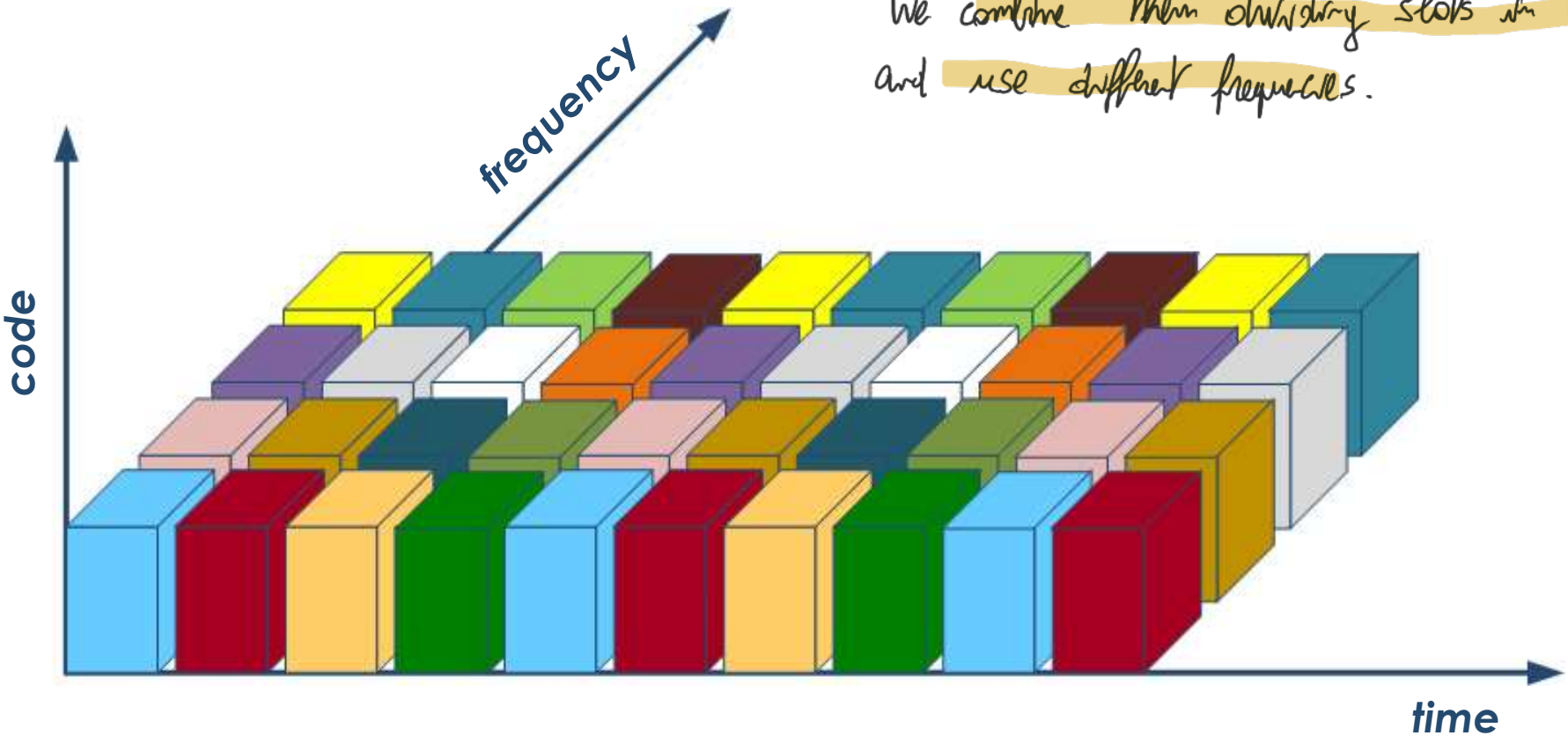


FDMA in 3D resource plan



Combining TDMA with FDMA: The GSM example

We combine them ordinary slots in time and use different frequencies.



Channelization: $\Delta_f = 200\,\text{kHz}$ **in** $[890 \div 915]\ \text{MHz}$, $N = 8$ **users**

Timing:

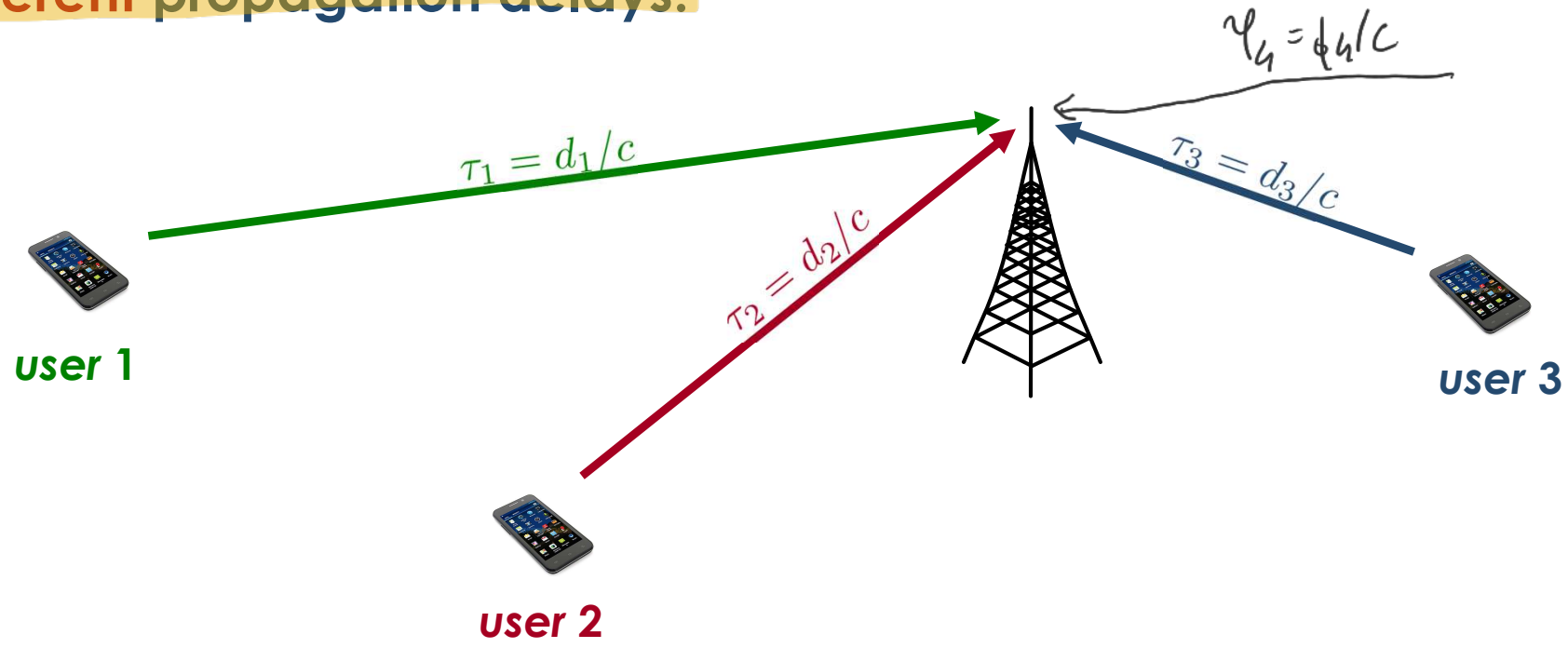
$$T/N \approxeq 3.69\,\mu\text{s}, \quad T_p = 148 \cdot T/N \approxeq 546.12\,\mu\text{s},$$
$$T_g = 8.25 \cdot T/N \approxeq 30.44\,\mu\text{s}$$

Code division multiple access (CDMA)

Code division multiple access (CDMA) (1/2)

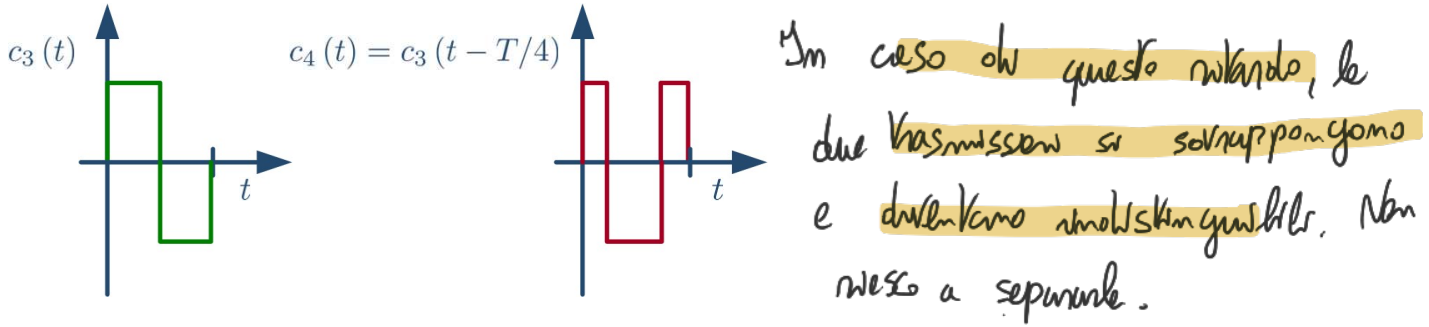
The main difference between CDM and CDMA is due to uncoordinated times of arrival across the network users

Similarly to the TDMA scenario, different users experience different propagation delays:



Code division multiple access (CDMA)* (2/2)

When considering orthogonal codes (such as the WH code set), cross-correlation is negligible if and only if $\Delta\tau \ll T/N = T_c$



This means that, in practice, the requirement is $\Delta\tau \leq T_c/10$

Example: in UMTS (3G), $T_c \approx 0.26\,\mu\text{s}$, which implies $\Delta\tau \leq 26\,\text{ns}$

This accuracy is not viable, especially if compared with that required by TDMA, in which $\Delta\tau \leq T_g \approx T$ (typically, accuracy is two-three orders of magnitude higher in CDMA)

Non posso usare W-H codes per questo motivo. Comunque, fin quando la differenza tra le sfasature è $< T_c/10$ va bene, ma in casi reali i requisiti di accuracy sono troppo stringenti. Oggi con tecnologie ancora più recenti la situazione è peggio.

CROSS CORRELATION FUNCTION measures the correlation between different signals:

$$R_{n,m}(0) = \frac{1}{T}\int_0^T C_n(t)\cdot C_m^*(t)\,dt = \delta[n-m] = \begin{cases} 1 & \text{if } n=m \\ 0 & \text{if } n\neq m \end{cases}$$

$R_{n,n}(0)=1 \qquad R_{n,m}(0)=0 \ \forall n\neq m.$

Now we want to compute $R_{n,m}(\Delta\varphi)$. Ideally we would want this function to always be 0: no matter the misalignment, we would want the two codes to not be correlated.

(1) In this case, you are able to understand when the bit received actually starts. This example works with GPS too. When I receive a signal from a satellite (satellites have associated a unique code with CDMA), I "scan" for the maximum of the correlation in the cross-correlation function because that will probably give me the exact frame in which the signal is sent.

In the example of the slide, C_3 and C_4 violate the cross correlation property for $n\neq m$ ($R_{n,m}(\Delta\varphi)=0 \ \forall\Delta\varphi$) because $R_{n,m}(T/4)=1$ because C_3 and C_4 are equal with a shift of $T/4$. Long story short: there is no code set that satisfies those two ideal conditions.

Non-orthogonal codes (1/4)

In this context, we need to identify signature codes that work pretty well in the asynchronous scenario

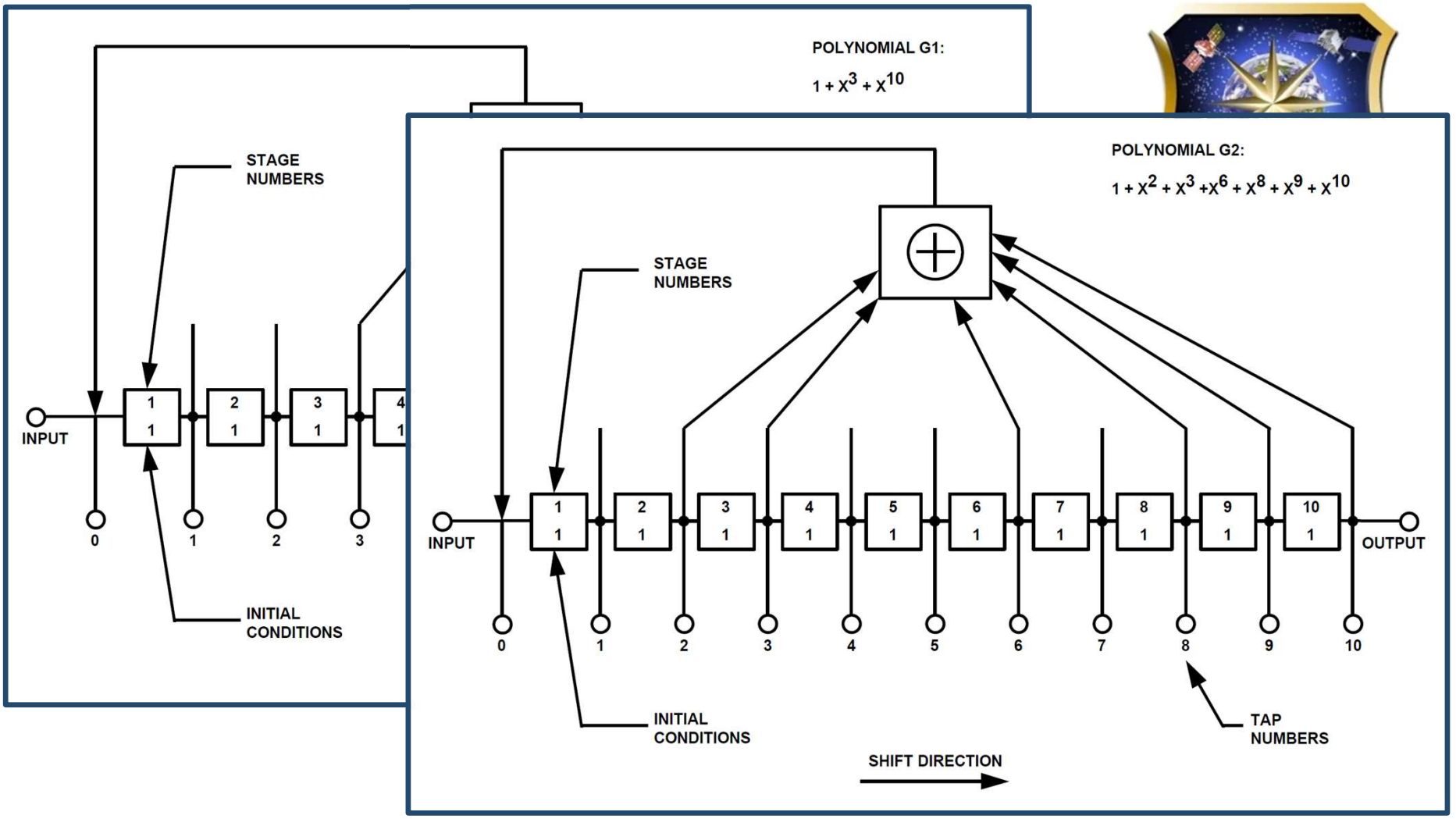
↳ if we observe transmissions without knowing they are transmitting they behave like noise.

The class of pseudo-random noise (PRN) sequences, such as the Gold codes used in many systems, provide good performance in fully uncoordinated transmission patterns

PRN codes show similarities with the AWGN (e.g., the DSP is almost flat in the frequency domain), due to the pseudo-random nature of the chips; however, they are generated according to a well-defined algorithm, known to both transmit and receive sides

Non-orthogonal codes (2/4)

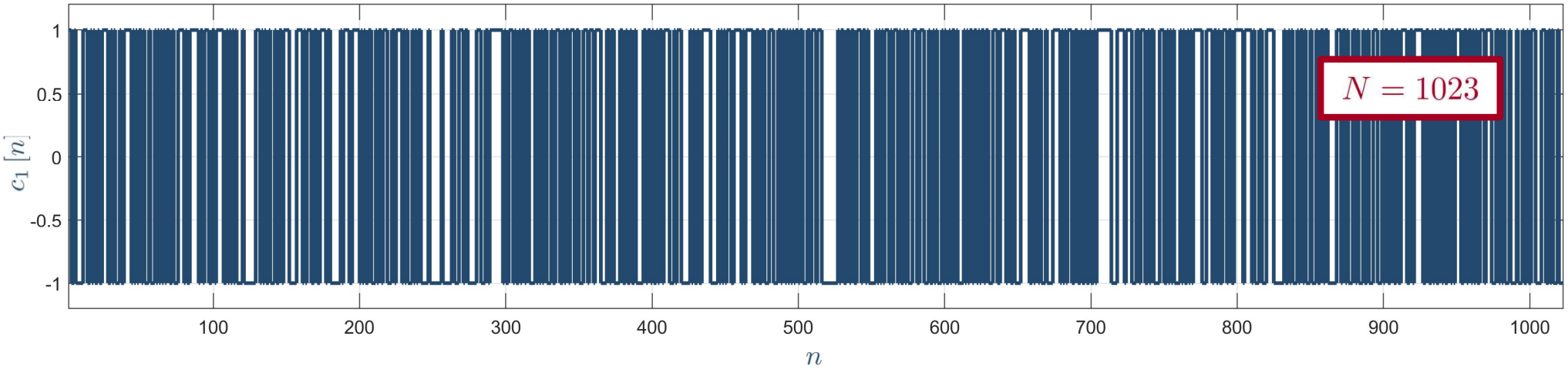
Example: PRN code generation in global positioning system (GPS)

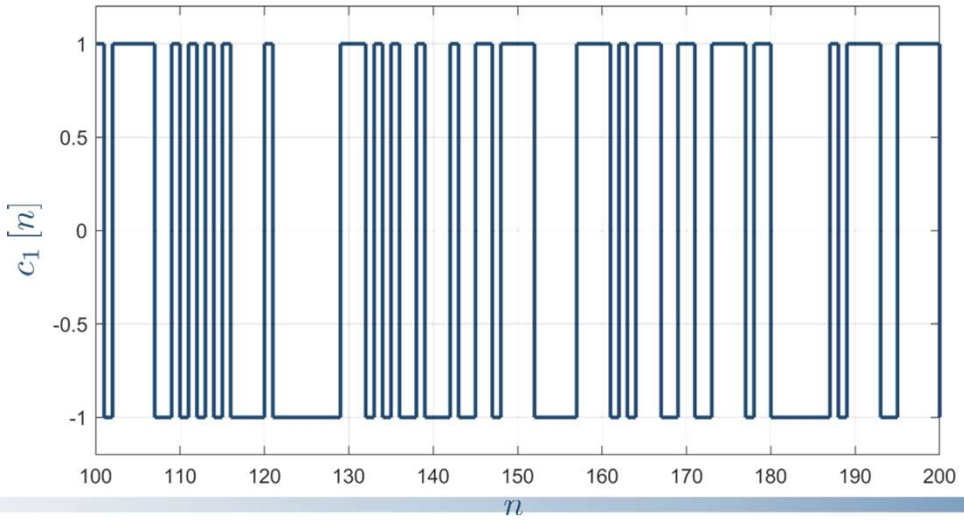


Non-orthogonal codes (3/4)

Example: PRN code generation in global positioning system (GPS)

GPS PRN ID 1

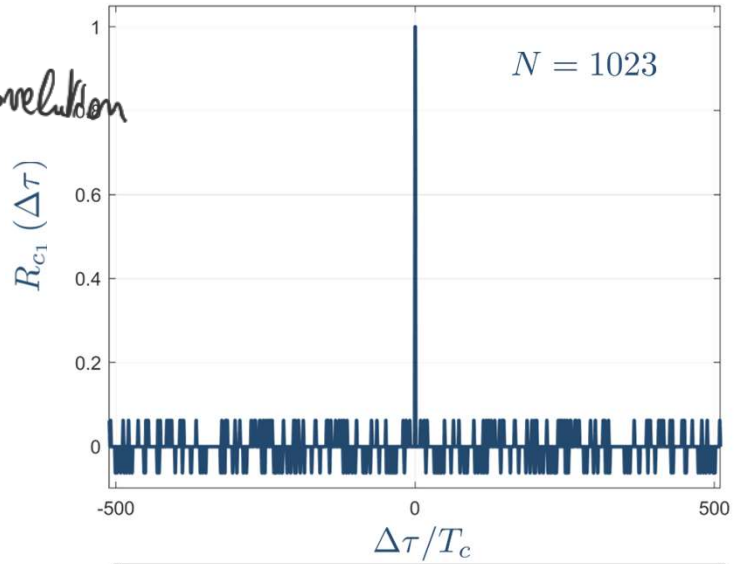


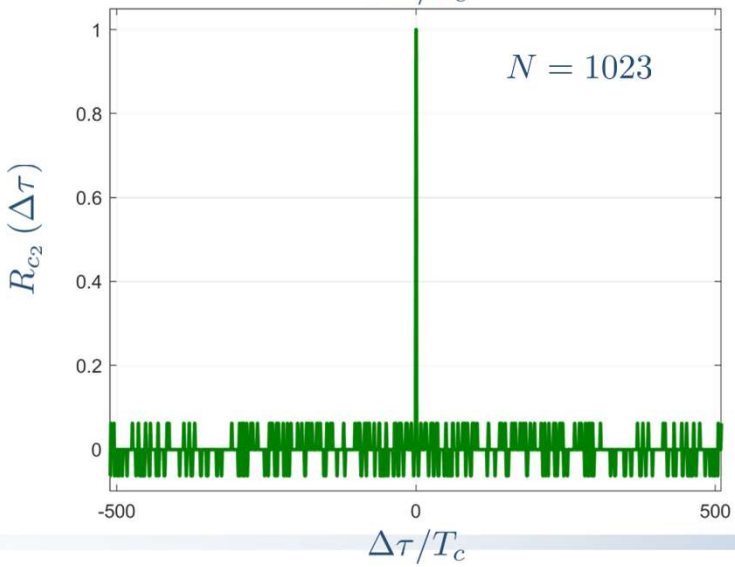


Non-orthogonal codes (4/4)

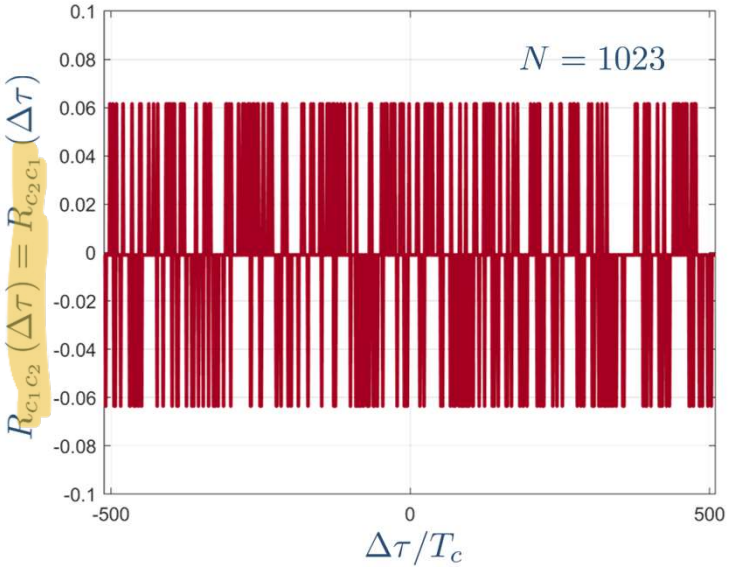
Example: PRN code generation in global positioning system (GPS)

Auto Correlation





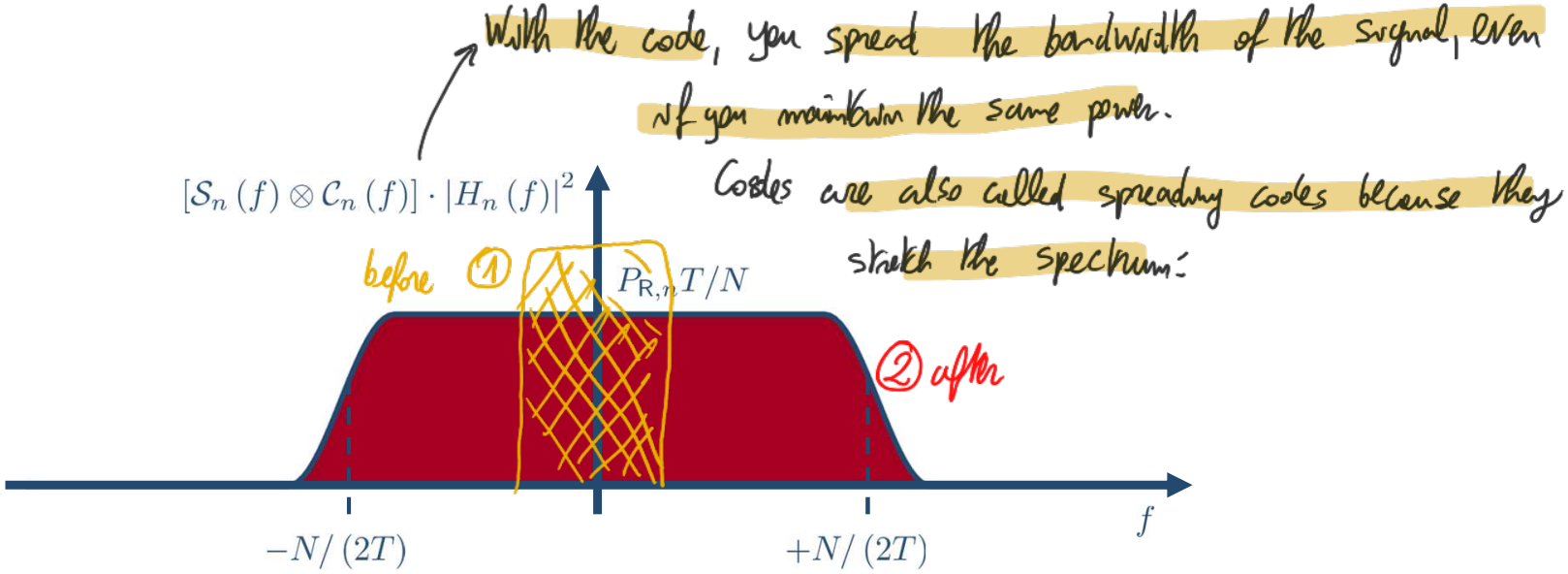
This is the best we can do:
For the autocorrelation, 1 in 0, close to 0 elsewhere
For cross correlation, 0 everywhere. This is acceptable



Cross Correlation

Performance of CDMA (1/4)

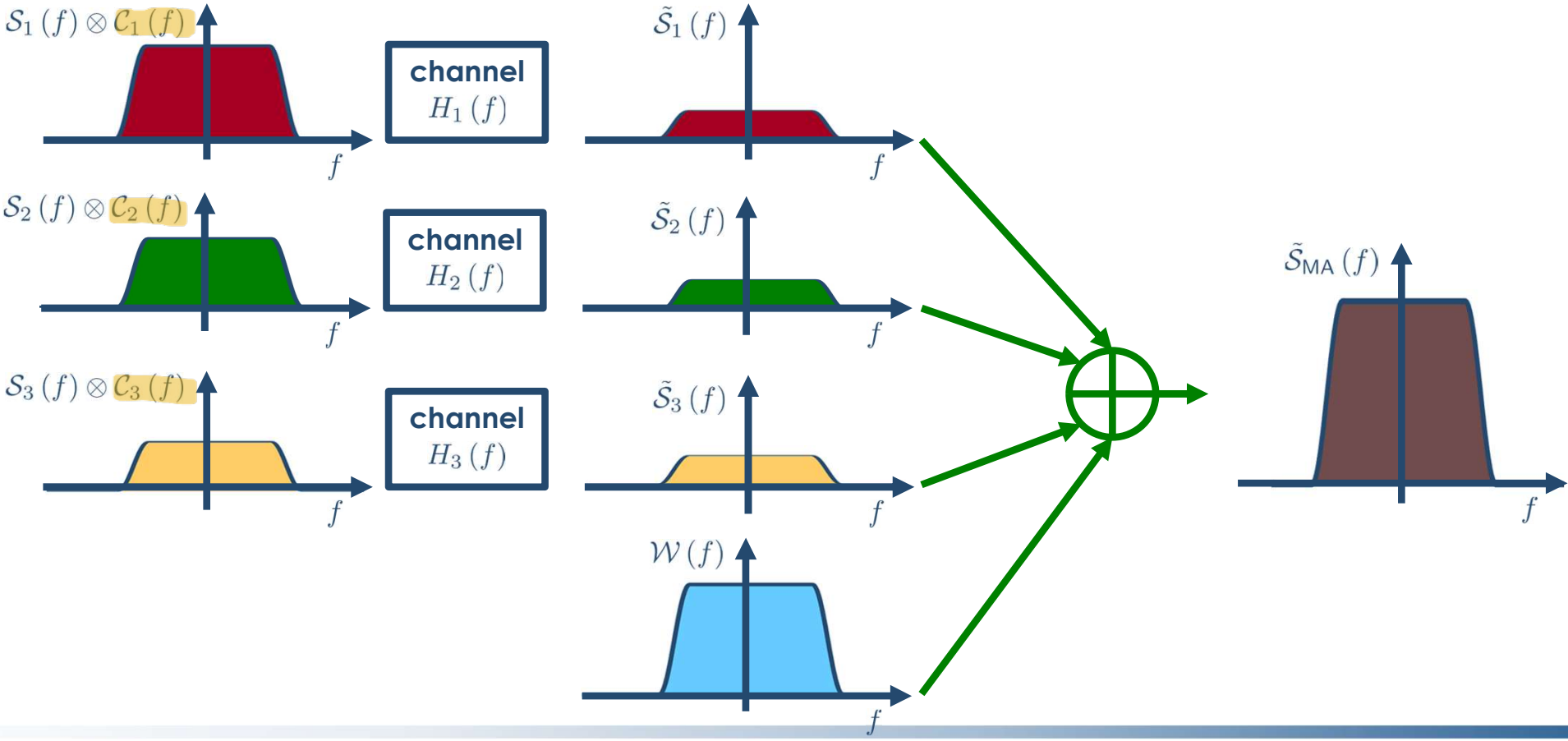
Thanks to the ad-hoc code chip generation, PRN codes show a flat PSD:



where $H_n(f)$ is the DFT of the channel experienced by user n, and $P_{R,n}$ is user n's received power

Performance of CDMA (2/4)

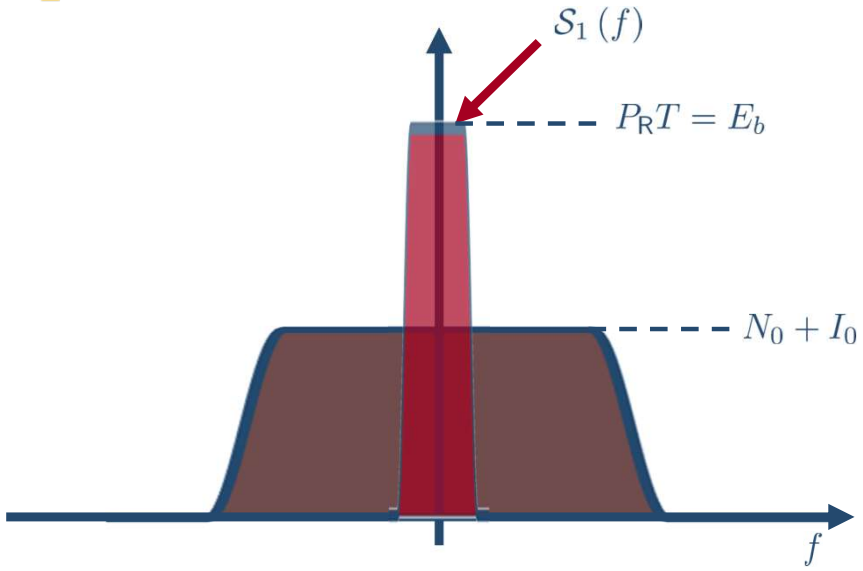
Hypothesis: our CDMA system adopts a closed-loop **power contol** that equalizes all **received** powers $\{P_{\mathrm{R},n}\}_{n=1}^{N_A} = P_{\mathrm{R}}$, where N_A is the number of users **simultaneously** active (note: $N_A \lessgtr N$)



Suppose user 1 is using code 1, user 2 with code 2 etc. Then the receiver sums everything together plus noise, so you have the brown signal. The receiver uses the same codes to multiply and extract the information.

Performance of CDMA (3/4)

Decoding the *n*th user by applying the matched-filter approach is equivalent to extract only the desired spectrum out of the compound signal:



Other users behave as noise with interference.
N_A = numbers of users in the system

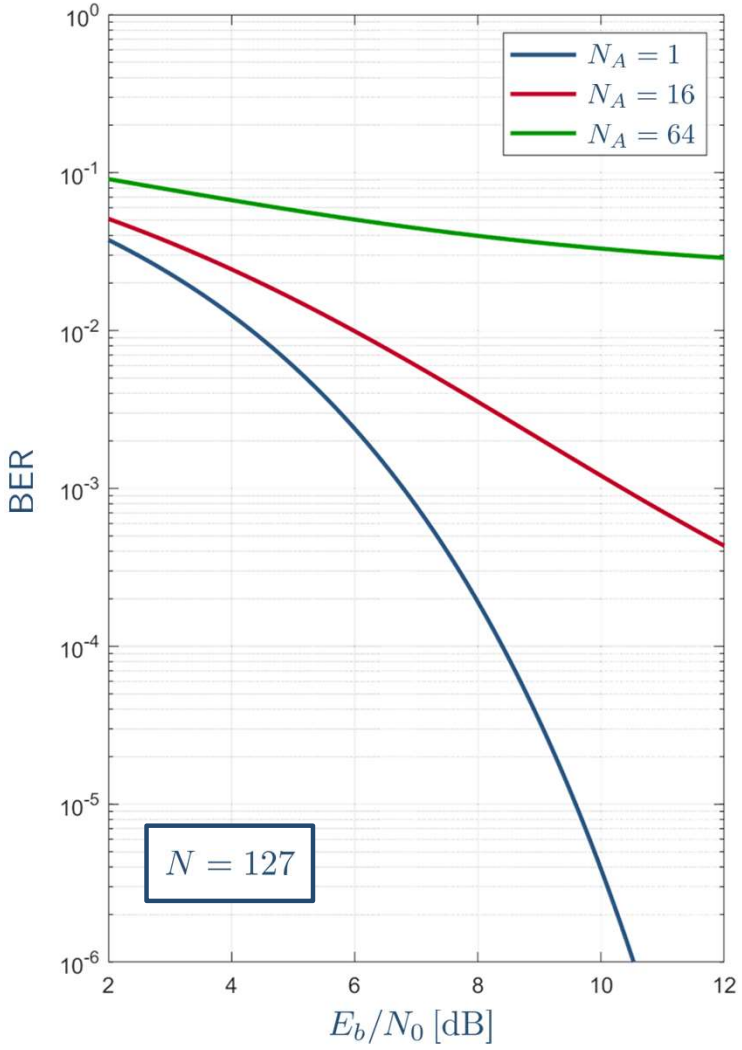
where the interference to the other user (the so-called multiple access interference, MAI) can be considered as a white Gaussian process as soon as $N_A \gg 1$ (say, $N_A \geq 10$)

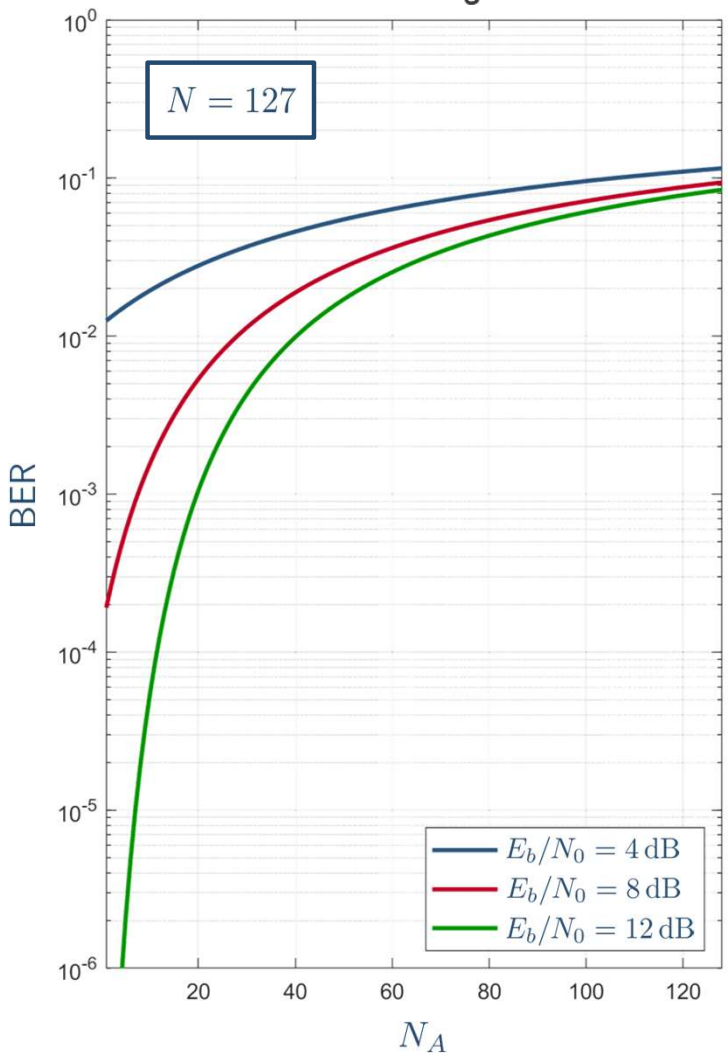
With CDM you use orthogonal codes with length N. Here the number of simultaneous streams $N_s \leq N$, otherwise you break orthogonality.

With CDMA you use non-orthogonal codes with length M, but now N_s can be $\geq M$, so we have no limitation over the number of users. The bigger N_s is, the more interference we have though.

Performance of CDMA* (4/4)

Plots done without channel coding





Differences wrt TDMA and FDMA

Non-orthogonality of the signature codes significantly degrades the performance of the network

However, CDMA allows users to access the network in a truly random manner, without the need for synchronization across the users

This places CDMA closer to statistical-based multiple-access techniques, such as packet-oriented ones

We can also adopt statistical multiple access, more decentralised.

JMP FDMA
SLIDE 95

LABEL

Packet-oriented multiple access

What is packet-oriented multiple access? (1/2)

Connection-oriented multiple access:

- **coordinated access**, such as in TDMA, FDMA, and, to some extent, CDMA
- **deterministic approach**, well suited for constant-bitrate communications

Packet-oriented multiple access:

- **uncoordinated access**
- **statistical approach**, well suited for bursty-traffic communications

What is packet-oriented multiple access? (2/2)

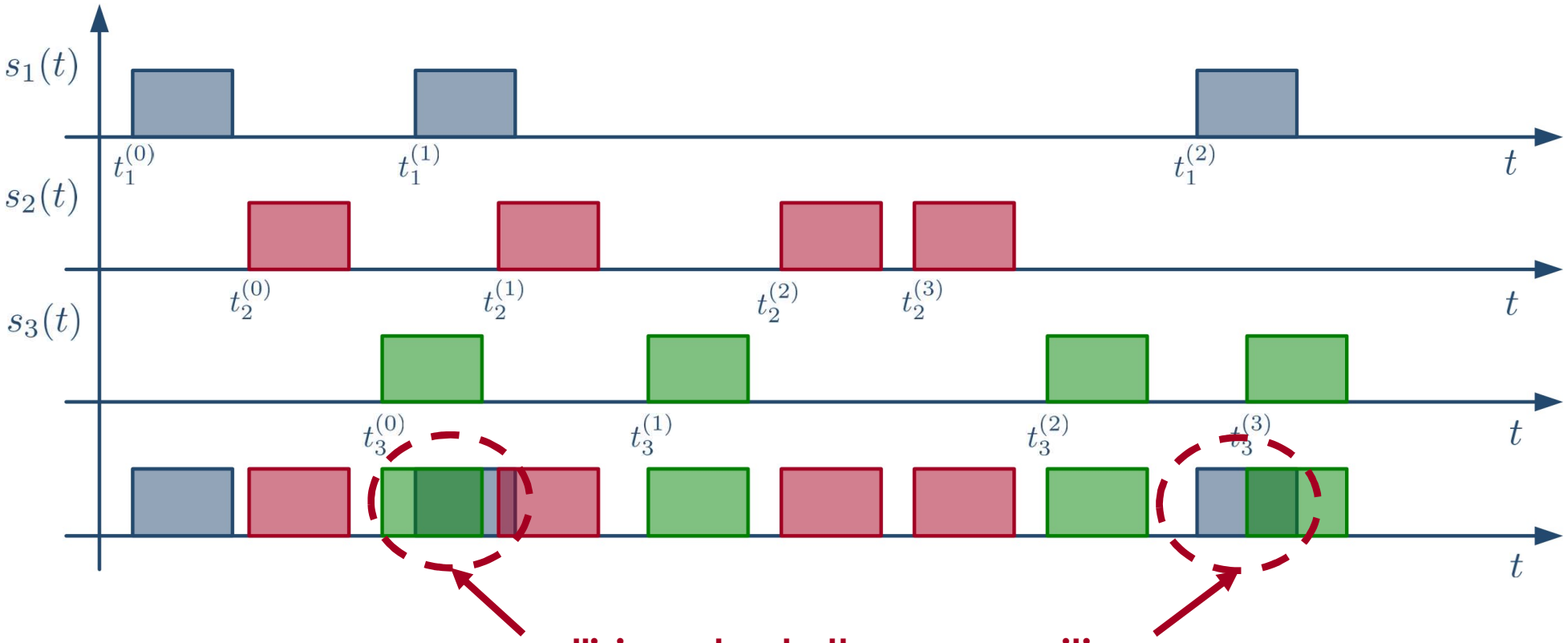
Typical example of bursty/impulsive traffic: the IoT scenario

The ALOHA protocol (1/3)

Historically, the first protocol to address this issue was the ALOHAnet, or simply **ALOHA**, devised in the early 1970s

The main goal was to connect the research centers at the University of Hawaii, spread across the Hawaiian islands, with the main Oahu campus

The ALOHA protocol (2/3)



collisions due to the superposition (in the time domain) of packets from different users

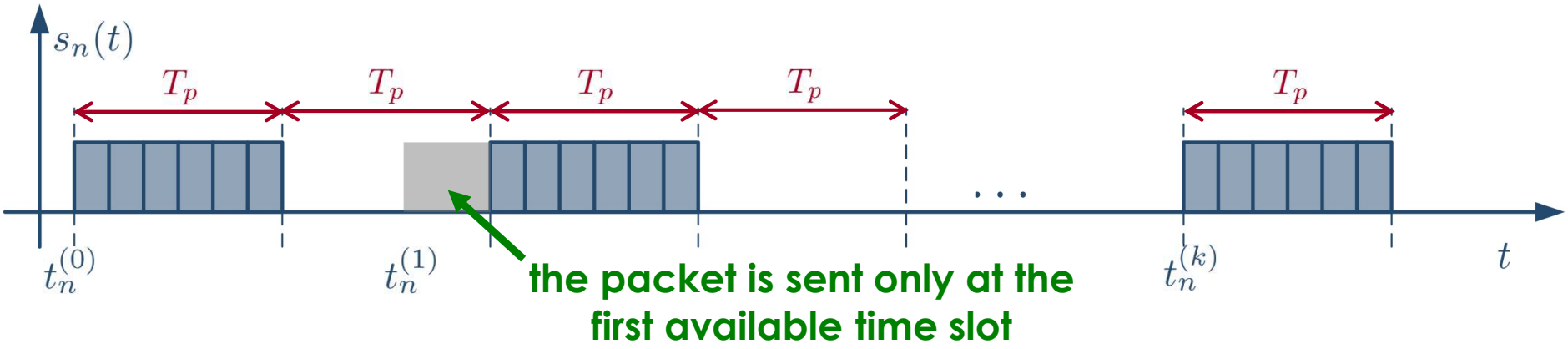
The ALOHA protocol (3/3)

Unlike connection-oriented multiple-access scheme, the ALOHA protocol needs a **return channel**, to receive feedback from the receiver (using **acknowledgment** of correct reception)

In case of a missing acknowledgement (corresponding to an event of collision), the **re-transmission** follows a specific backoff time (based on a statistical approach – out of scope in this lecture)

The slotted-ALOHA (S-ALOHA) protocol (1/2)

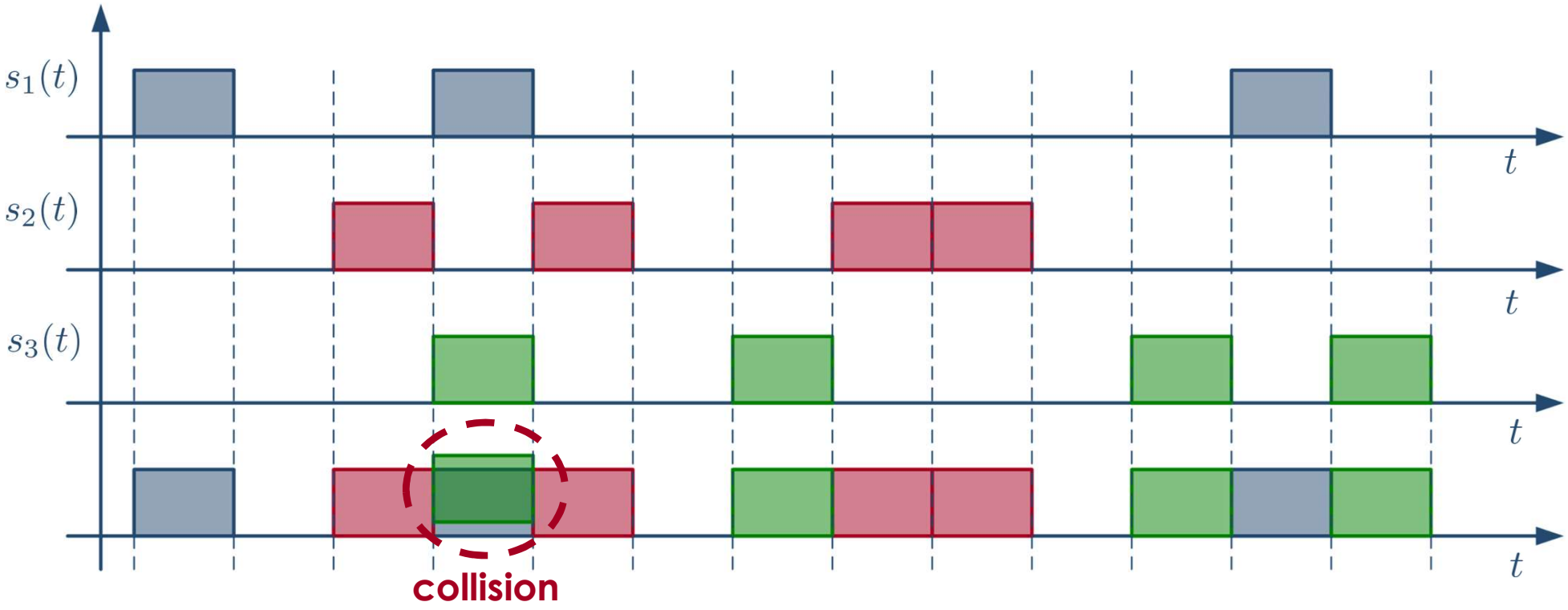
To reduce the occurrence of collision events, we can apply a centralized **network synchronization** similar to TDMA:



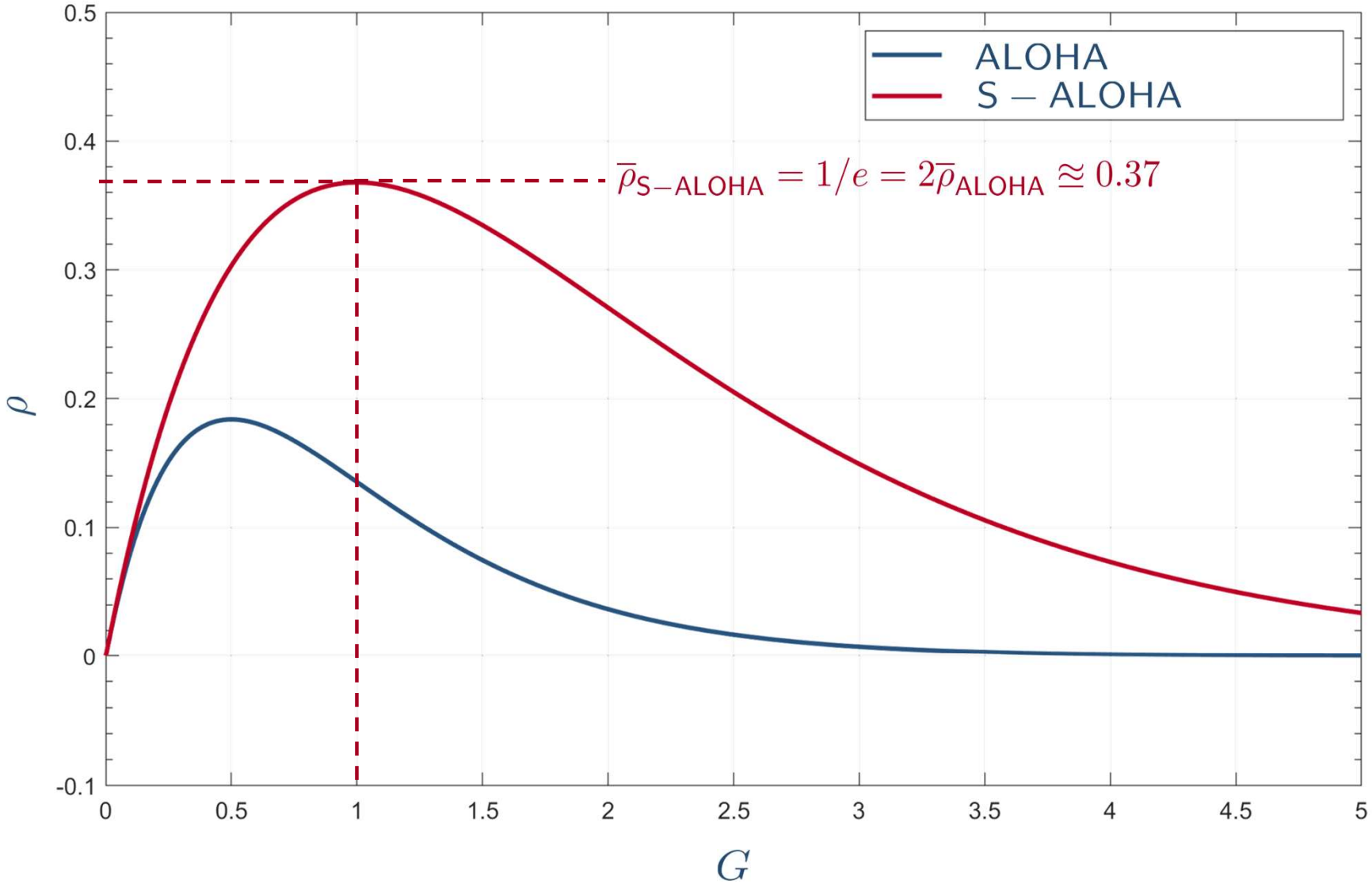
This modified version of ALOHA is called **slotted ALOHA (S-ALOHA)**

The slotted-ALOHA (S-ALOHA) protocol (2/2)

Collisions occur only when more than one source selects the same time slot to transmit the packet (i.e., partial collisions do not exist anymore)



ALOHA vs. S-ALOHA*



Bibliography

[01] B. Sklar and F. Harris, *Digital Communications* , 3rd ed. Upper Saddle River, NJ, US: Prentice Hall, 2021.

[02] J.G. Proakis and M. Salehi, *Digital Communications,* 5th ed. New York, NY: McGraw-Hill, 2007.

[03] M. Luise, *The wireless communications engineer's toolkit*. Pisa, Italy, Oct. 2021. [Online] http://www.iet.unipi.it/m.luise/Signals_Systems.pdf

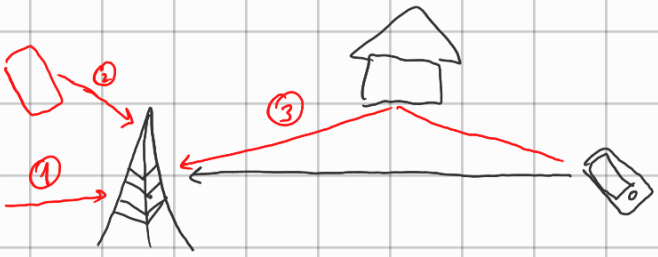
[04] H. Benoit, *Digital Television: Satellite, Cable, Terrestrial, IPTV, Mobile TV in the DVB Framework*, 3rd ed. Burlington, MA: Elsevier, 2008.

[05] T.S. Rappaport, *Wireless Communications: Principles and Practice,* 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2002.

[06] A.J. Goldsmith, *Wireless Communications*. Cambridge, UK: Cambridge Univ. Press, 2005.

[07] A.F. Molisch, *Wireless Communications*. West Sussex, UK: J. Wiley & Sons, 2005.

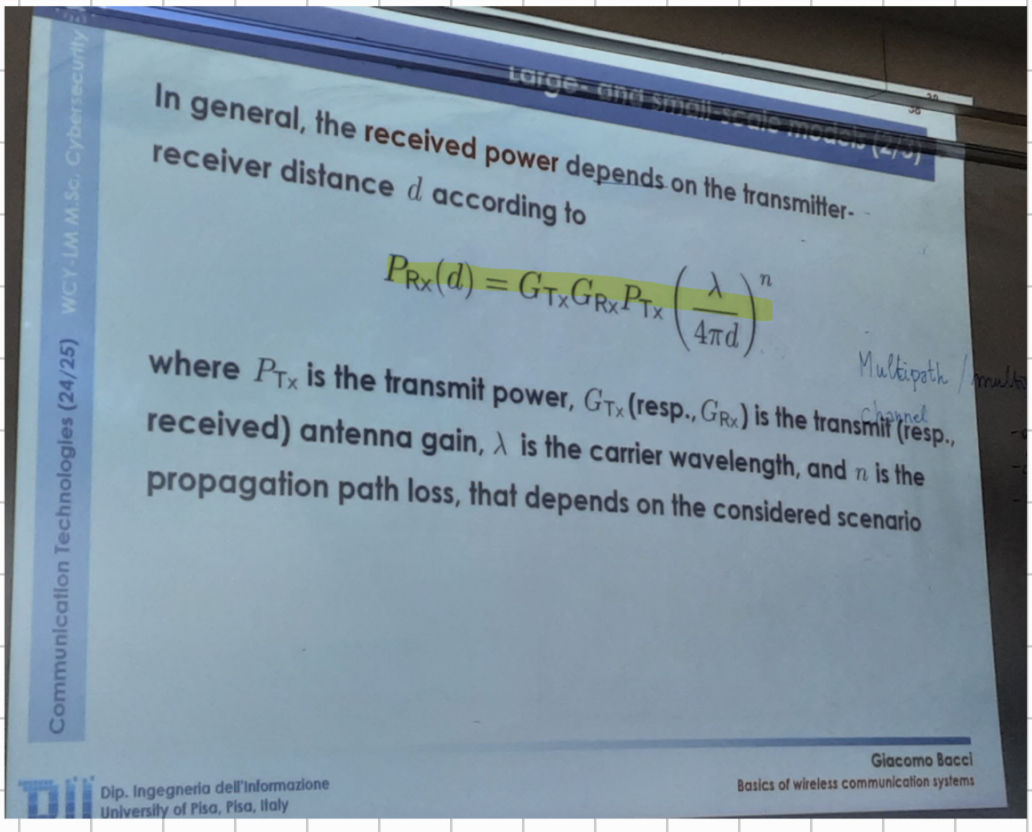
Now we focus on Wireless channels: we need to understand the specific propagation conditions to use a wireless channel.



Base transceiver station

Comm. between mobile station and base station is called uplink. Opposite is downlink.
Ideally we just communicate, but we have ① noise and ② multiple access interference and ③ environment obstacles. We have multiple copies of the same signal for ③.
This channel, with those elements is called multipath channel (obstacles).

We have to deal with attenuation, noise, distortion (3)



$P_{Rx}(d)$ = received power, function of the transmission power, G_{Tx}, G_{Rx}, the gains of the antennas (let's assume they are 1 for now) and other terms.

If d increases, the received power is lower. This models the attenuation. But not just that: when we increase the carrier frequency, the carrier wavelength λ decreases, so we have more attenuation. Ideally, to combat the noise, we want a high receiver power.

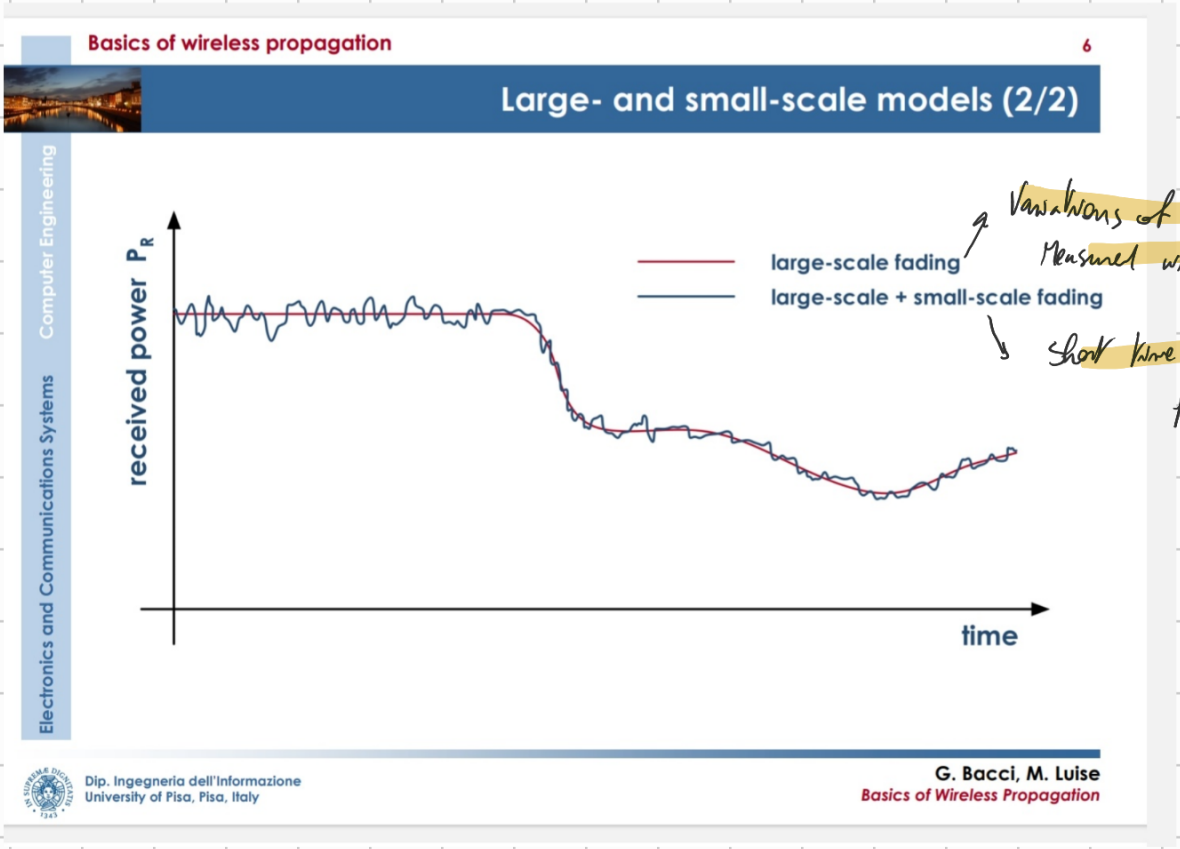
So, if $f_0 \uparrow \Rightarrow P_{Rx}(d) \downarrow$ BAD
if $d \uparrow \Rightarrow P_{Rx}(d) \downarrow$ BAD

$$P_{Rx}(d) = G_{Tx} G_{Rx} P_{Tx} \left(\frac{\lambda}{4\pi d}\right)^2$$

(Attenuazione del percorso)

In general we have n = path loss exponent. If n gets large, the received power decreases (in general $\lambda/4\pi d < 1$). Without any obstacle in free space $n=2$. In a city, for example, a reasonable value is $n=2.5$. All the obstacles absorb the signal and take some part of the power. $n \geq 2$.
Best case $n=2$.

Small scale fading measures the microscopic fluctuations of received signal.



Variations of significance. Measured with mean value

Short time period fluctuations

Large- and small-scale models (1/2)

A better suited propagation model is composed by:

- **large-scale models**, that predict the average energy received in a wireless system as a function of the distance between the transmitter and the receiver
- **small-scale models**, that account for the instantaneous variations in the propagation conditions

LEZIONE 11 END

What are large scale and small scale models in wireless channels?

In wireless communications, **large-scale** and **small-scale** models describe the characteristics of radio signal propagation over different spatial and temporal scales:

Large-Scale Models

These models describe the average signal power behavior over long distances, typically several wavelengths or beyond. They focus on the overall trend of signal attenuation and account for factors such as:

- **Path Loss**: The reduction in signal power as a function of the distance between the transmitter and receiver.
- **Shadowing (or Slow Fading)**: Signal variations caused by large obstacles like buildings or hills that block or scatter the signal.

Large-scale models are used to predict signal strength over a wide area and are crucial for tasks like network planning and cell coverage estimation.

Small-Scale Models

These models describe rapid fluctuations of the signal over short distances (comparable to a few wavelengths) or short time intervals. These fluctuations arise from:

- **Multipath Propagation**: Multiple copies of the transmitted signal arrive at the receiver via different paths, causing interference (constructive or destructive).
- **Doppler Effect**: Frequency shifts due to relative motion between the transmitter and receiver.

Small-scale models are important for understanding and predicting signal variations in real-time, especially in dynamic environments like urban areas or when users are mobile.

Relationship Between the Two

- **Large-scale effects** give the general trend of signal strength over large areas or distances.
- **Small-scale effects** overlay rapid, local variations onto the large-scale trend.

Together, these models provide a comprehensive understanding of wireless channel behavior.