Compressione
Parte 1

Prof. Filippo Milotta
milotta@dmi.unict.it

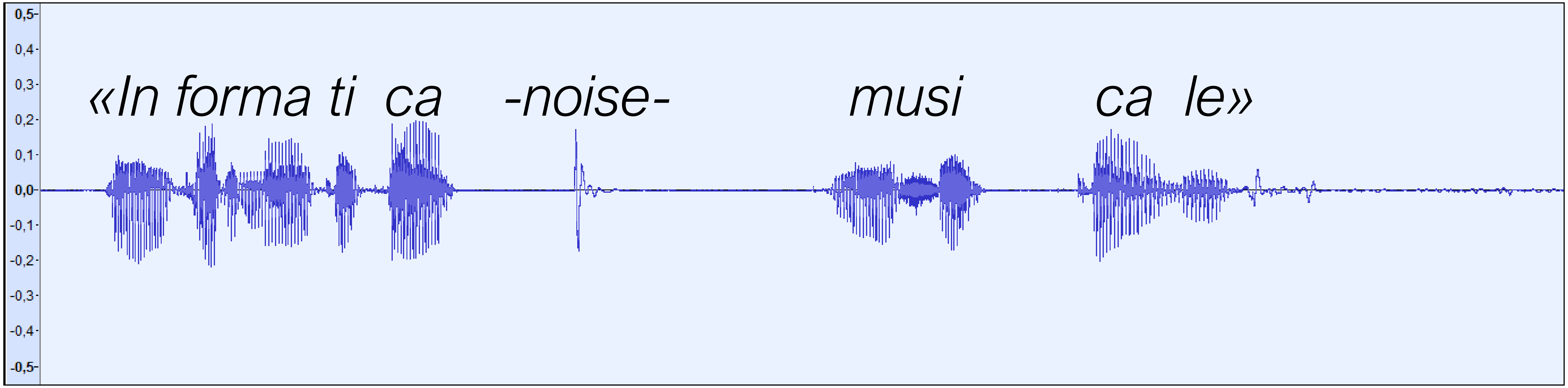
Perché comprimere?

1. Riduzione dello spazio di memoria occupato

2. Riduzione dei tempi (e costi) di trasmissione

Compressione…

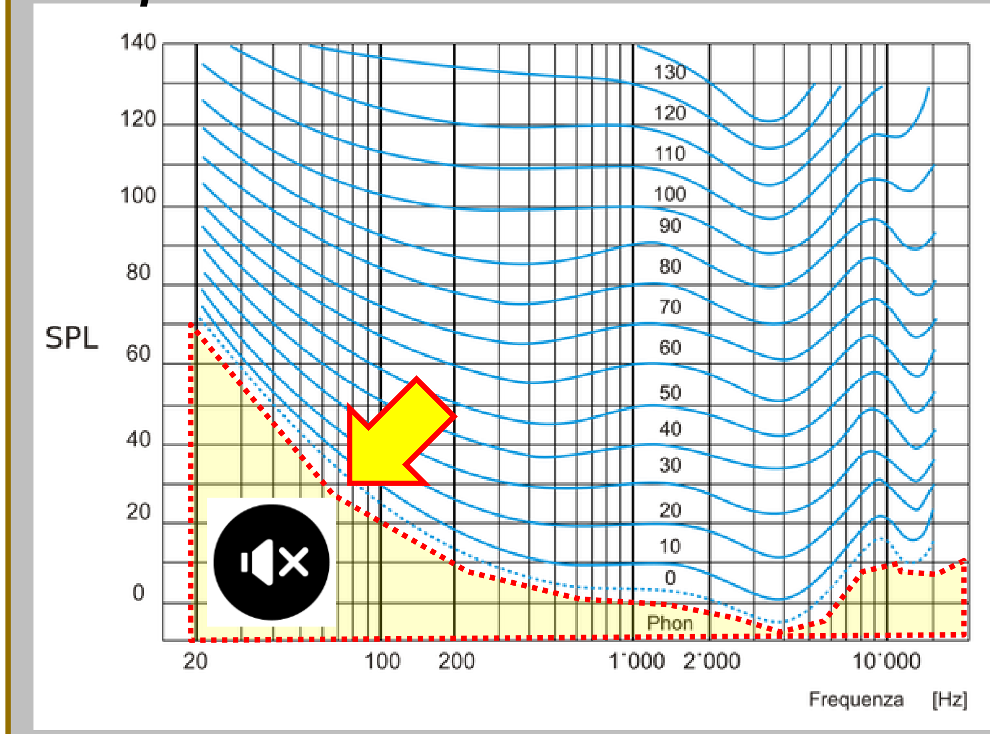
- Come comprimere una traccia del genere?



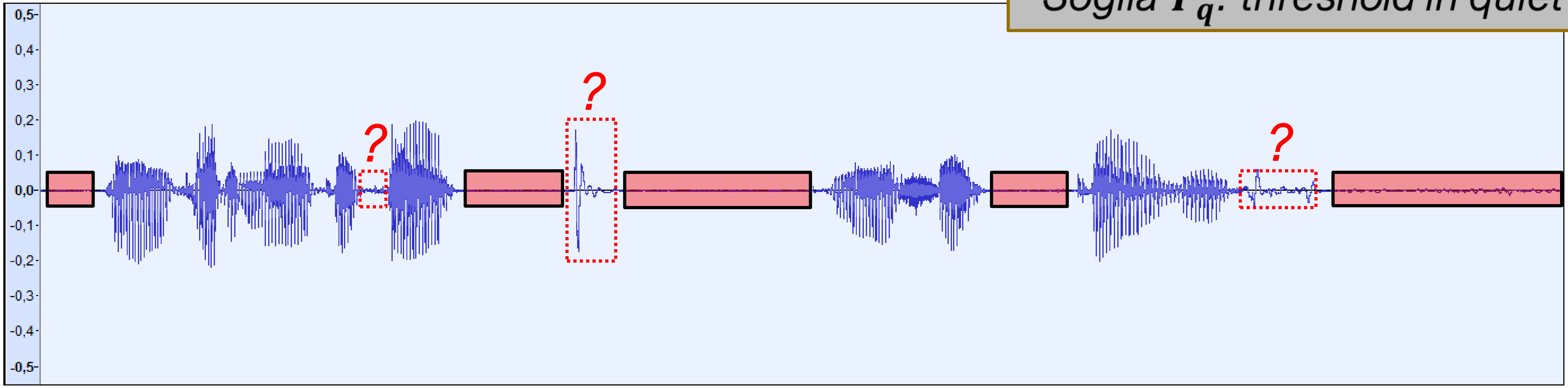
Compressione del silenzio (Metodo Naïve)

- Tecnica Lossy
  - Soglia di intensità sonora
  - Soglia di attivazione temporale

Ripasso



Soglia T_q: threshold in quiet



Audio digitale – Spazio in memoria

Sia $\boldsymbol{f_c}$ il tasso di campionamento , $\boldsymbol{N}$ la profondità in bit, $\boldsymbol{D}$ la durata del flusso audio e $\boldsymbol{C}$ il numero di canali, allora il numero di bit necessari a rappresentare il segnale (senza compressione) si calcola:

$$Size = f_c \times N \times D \times C$$

Il numero di bit che fluisce nell'unità di tempo (un secondo) prende il nome di bit rate. Si misura in bps (bit per secondo).

$$bitrate = f_c \times N \times C$$

E' chiaro che l'obiettivo è garantire una buona qualità utilizzando la minima quantità di memoria. I metodi di compressione rappresentano un passo successivo che permette di abbassare il **bit rate** preservando la qualità.

Audio digitale – Spazio in memoria
Un esempio pratico

- CD Audio
  - Tasso di campionamento: 44,1kHz
  - Profondità in bit: PCM lineare 16 bit
    - Bitrate → 44,1kHz * 16 = 705,6kbps
  - Canali: Segnale stereo (2)
    - Bitrate → 705,6kbps * 2 = 1.411Mbps
  - 1 Minuto di registrazione: 44100*16*2*60/8 ~ 10MB

Codifiche μ-law e A-law

A-law e μ-law sono due algoritmi di codifica definiti nello standard **G.711**

- Sono stati pensati per codificare digitalmente e trasmettere su una rete ISDN, la **voce al telefono**. Si parla quindi di suoni con frequenza da 0 a $4KHz$.

- Per ottimizzare l'utilizzo di banda, le due codifiche usano 8000 campioni al secondo ($4\,KHz \times 2$) e una **quantizzazione non uniforme (logaritmica)** a 8 bit. Il bit rate sarà dunque di $64\,Kbps$ (8000×8), esattamente la larghezza di banda di una rete ISDN (**Integrated Services Digital Network**).

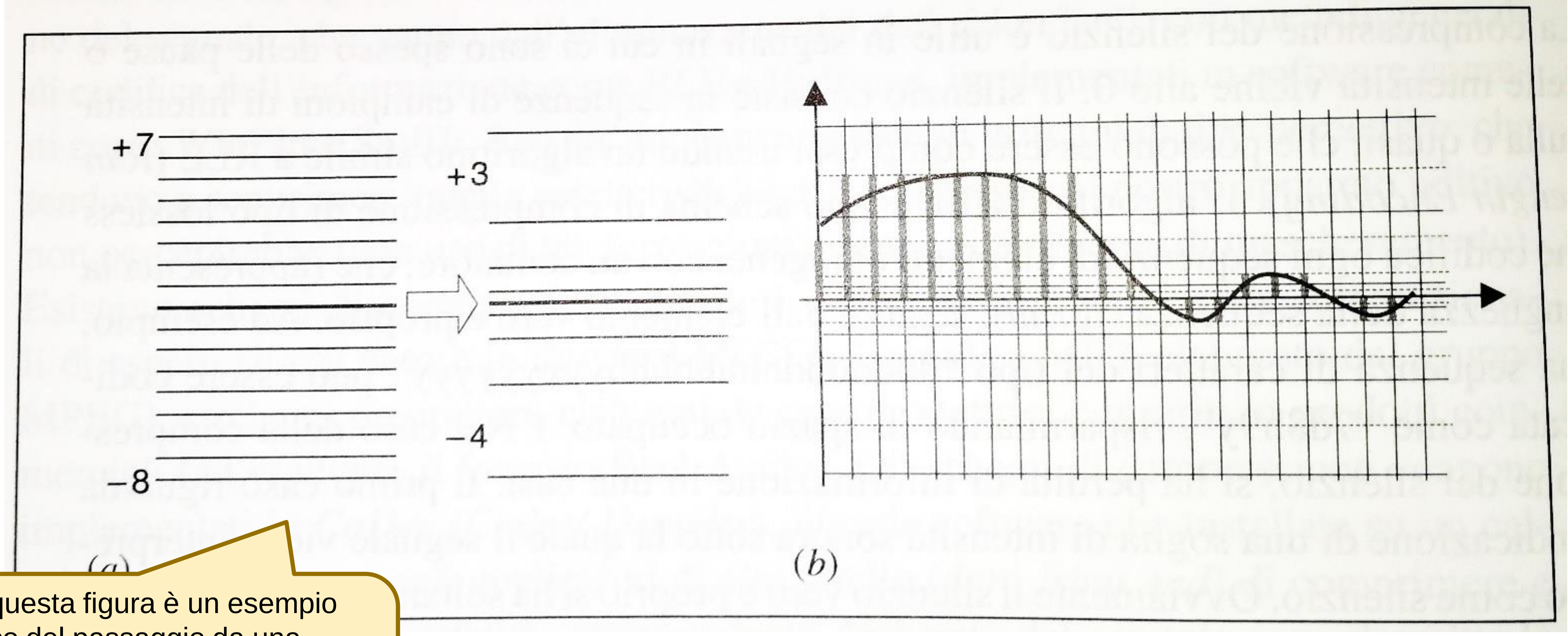
- La quantizzazione non lineare assicura maggiore precisione alle ampiezze più basse, garantendo una buona qualità con soli 8 bit.

Codifica μ-law

Partendo da campioni a 16 bit

La codifica ***μ*-law** è in uso in Nord America e Giappone. Grazie alla quantizzazione non lineare, permette di ottenere con soli 8 bit la stessa qualità (es: SQNR) che si otterrebbe con una quantizzazione lineare a 14 bit.

I valori di Y attorno allo zero (più piccoli in valore assoluto) sono quelli a cui saranno dedicati più bit.



Nota Bene: questa figura è un esempio schematico del passaggio da una quantizzazione uniforme ad una non uniforme. Anziché passare da 16 bit a 8, in questa figura passiamo solo da 4 a 3

Codifica μ-law

Con 16 bit posso rappresentare 65536 valori, da -32768 a 32767, compreso lo 0

$$\mu = 255$$

$$-32.768 \leq x \leq 32.767$$

$$(Normalizzata:) -1 \leq x \leq 1$$

$$0 \leq Y \leq 255$$

$$\mathrm{Y} = \begin{cases} 128 + \boxed{\underbrace{\boxed{\underbrace{\frac{127}{ln(1+\mu)}}_{22,9}} \times \boxed{\underbrace{ln(1+\mu|x|)}_{[0;\,5,5]}}}_{[0;\,127]}} & x \geq 0 \\ 127 - \frac{127}{ln(1+\mu)} \times ln(1+\mu|x|) & x < 0 \end{cases}$$

Codifica µ-law

$$Y = \begin{cases} 128 + \dfrac{127}{ln(1+\mu)} \times ln(1+\mu|x|) & x \geq 0 \\ 127 - \dfrac{127}{ln(1+\mu)} \times ln(1+\mu|x|) & x < 0 \end{cases}$$

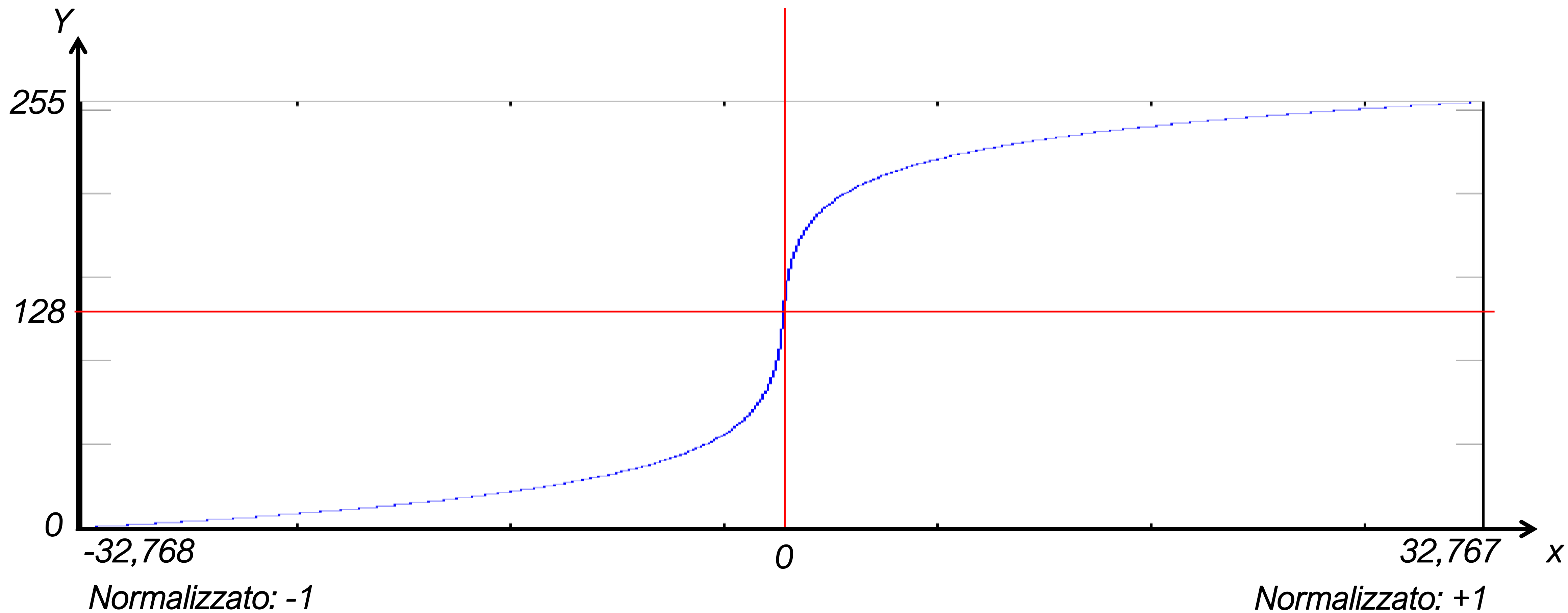
- Questa formula **comprime** campioni a 16 bit con segno in modo non lineare su campioni da 8 bit senza segno (da 0 a 255)

$$\mu = 255$$
$$-32.768 \leq x \leq 32.767$$
$$0 \leq Y \leq 255$$

I valori di Y attorno allo zero (più piccoli in valore assoluto) sono quelli a cui saranno dedicati più bit.

Esempio μ-law

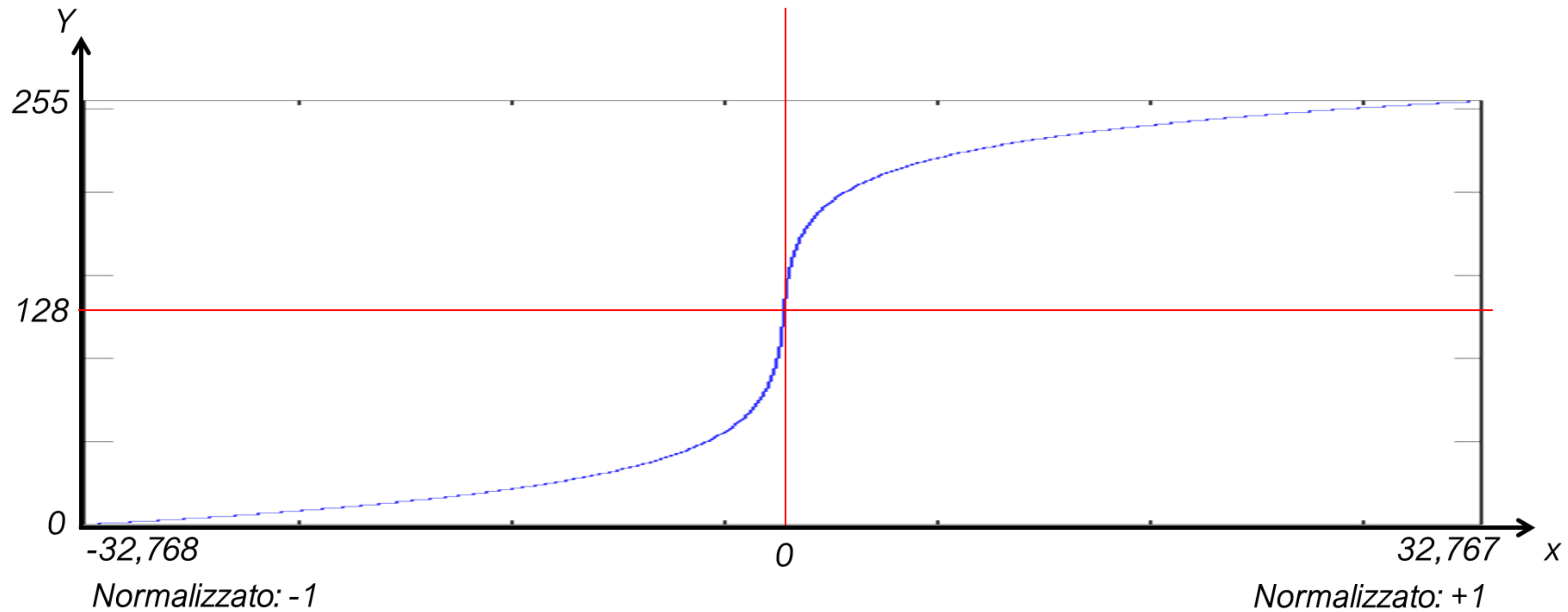
$$Y = \begin{cases} 128 + \dfrac{127}{ln(1+\mu)} \times ln(1+\mu|x|) & x \geq 0 \\ 127 - \dfrac{127}{ln(1+\mu)} \times ln(1+\mu|x|) & x < 0 \end{cases}$$



Sull'asse orizzontale sono presenti valori a 16 bit (interi tra 0 e 65535). Sull'asse verticale si trovano i corrispondenti interi a 8 bit ottenuti con la codifica μ-law. Si noti come i valori agli estremi siano quantizzati in maniera meno precisa.

Esempio µ-law

$$Y = \begin{cases} 128 + \dfrac{127}{ln(1+\mu)} \times ln(1+\mu|x|) & x \geq 0 \\ 127 - \dfrac{127}{ln(1+\mu)} \times ln(1+\mu|x|) & x < 0 \end{cases}$$



Campione originale	Nuovo campione
-32768	0
…	…
-32100	0
-32000	1
…	…
-200	106
-100	113
…	…
0	128
…	…
100	141
200	149
…	…
32000	254
32100	255
…	…
32767	255

Sull’asse orizzontale sono presenti valori a 16 bit (interi tra 0 e 65535). Sull’asse verticale si trovano i corrispondenti interi a 8 bit ottenuti con la codifica µ-law. Si noti come i valori agli estremi siano quantizzati in maniera meno precisa.

Decodifica μ-law

$$\mu = 255$$
$$-32.768 \leq x \leq 32.767$$
$$(Normalizzata:) -1 \leq x \leq 1$$
$$0 \leq Y \leq 255$$

CAVEAT - NOTA:
Nel libro la formula riporta un errore. Questa è la formula corretta!

$$x = \begin{cases} \dfrac{exp\left(\dfrac{Y-128}{127} \times \underbrace{\boxed{\ln(1+\mu)}}_{5,5}\right) - 1}{\mu} & Y \geq 128 \\ -\dfrac{exp\left(\dfrac{127-Y}{127} \times \ln(1+\mu)\right) - 1}{\mu} & Y < 128 \end{cases}$$

Quanto vale x se Y = 0?
Quante vale x se Y = 255?

Codifica μ-law

- E’ lossy o lossless?
  - Perché?
  - Verificare la risposta data applicando le formule

A-law

La codifica **A-law** è in uso in Europa. Grazie alla quantizzazione non lineare, permette di ottenere con soli 8 bit la stessa qualità (es: SQNR) che si otterrebbe con una quantizzazione lineare a 13 bit.

Sia X il valore originale di ampiezza **normalizzato** tra [-1,1], A un fattore pari a 87.7 (o 87.6), allora il valore codificato Y normalizzato in [-1,1] si calcola:

$$Y = \mathrm{sign}(X) \begin{cases} \dfrac{A|X|}{1+\ln A} & |X| < \frac{1}{A} \\ \dfrac{1+\ln A|X|}{1+\ln A} & \dfrac{1}{A} < |X| \leq 1 \end{cases}$$

Quanto vale Y se X = 1?
Quanto vale Y se X = -1?
Attenzione a non sbagliare formula, gli intervalli della X sono intesi con il valore assoluto!

2019 20

A-law decodifica

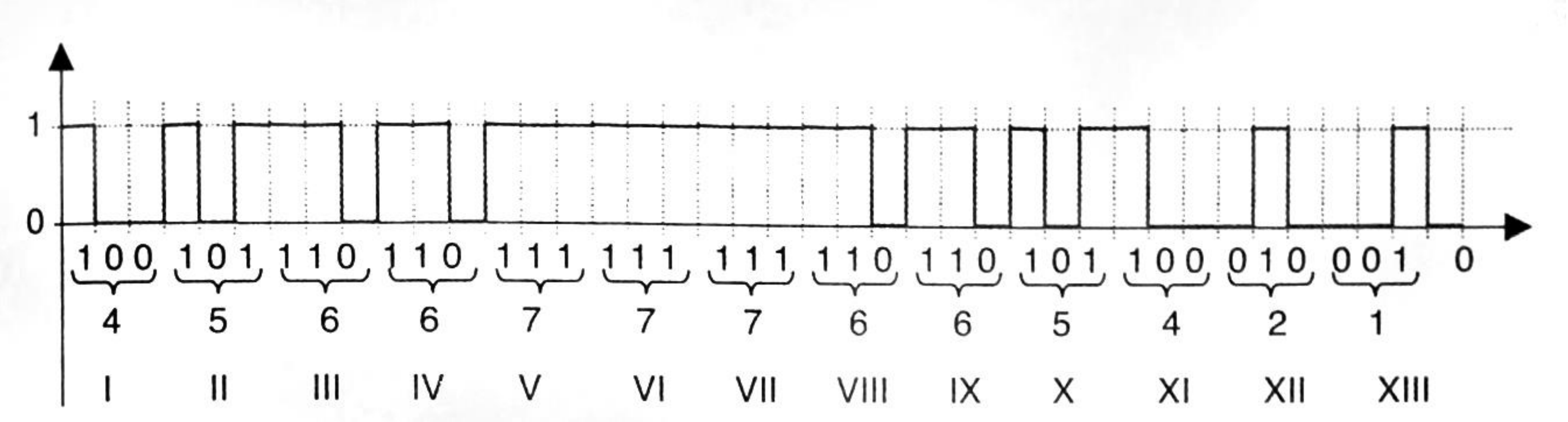
Sia $\boldsymbol{Y}$ il valore codificato di ampiezza **normalizzato** tra [-1,1], $\boldsymbol{A}$ un fattore pari a 87.7(o 87.6), allora il valore decodificato $\boldsymbol{X}$ normalizzato in [-1,1] si può riottenere dalla seguente legge:

$$X = \mathrm{sign}(Y)\begin{cases} \dfrac{|Y|(1+\ln A)}{A} & |Y| < \dfrac{1}{1+\ln A} \\ \dfrac{e^{|Y|(1+\ln A)-1}}{A} & \dfrac{1}{1+\ln A} < |Y| \le 1 \end{cases}$$

Tutte le considerazioni sulla normalizzazione fatte per la codifica μ-law, valgono pure per A-law. Per entrambe le codifiche possono essere definite delle tabelle di conversione per passare da codeword di <u>14</u> a 8 bit (μ-law) e da <u>13</u> bit a 8 bit (A-law).

Codifica PCM

La **Pulse Code Modulation** (PCM), è forse la più semplice tecnica di codifica di un audio digitale. In effetti non si fa altro che considerare ogni singolo campione come un impulso e associarvi una parola binaria che ne rappresenta l'ampiezza. La lunghezza delle parole binarie dipende ovviamente dai bit di quantizzazione (lineare) utilizzati.



Nell'esempio si può osservare la codifica PCM a 3 bit di un segnale audio. I 13 campioni assumono valori tra 0 e 7.

Codifiche DPCM e ADPCM

La **Differential Pulse Code Modulation** (DPCM), è una versione della PCM pensata per comprimere in maniera lossless. Anziché codificare i valori di ampiezza, sia codificano solo le differenze. Si tratta di una semplice codifica **differenziale**.

La **Adaptive Differential Pulse Code Modulation** (ADPCM), è una tecnica di codifica più sofisticata della DPCM. Oltre a codificare le differenze utilizza un meccanismo di predizione unito ad un algoritmo di riquantizzazione che si «**adatta**» alle differenze da codificare. Brevemente diciamo che riquantizza le differenze più grandi tra valori reali e valori predetti.

Differencing in DPCM

- *Differencing*:

…	100	101	102	103	103	103	102	101	…
…	…	+1	+1	+1	0	0	-1	-1	…

- In caso di differenze ridotte c'è un vantaggio effettivo nel codificare le differenze piuttosto che le codeword stesse
  - Si possono utilizzare delle LUT (Look-Up Table) (cioè delle tabelle-dizionario)

Predizione in ADPCM

- Per predire il campione successivo si somma ±1 al campione predetto attuale
  - Si sceglie la predizione più vicina al valore effettivo per codificare la differenza
  - Scelta diversa da ±1 ?

In questo esempio, in caso di dubbio (ambiguità) si sceglie sempre il valore predetto[n-1]+1

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Valore Effettivo	100	101	102	103	103	103	102	101	100	100	101	101	100	97
Predetto[n-1]+1		$\boxed{101}$	$\boxed{102}$	$\boxed{103}$	$\boxed{104}$	105	104	103	102	$\boxed{101}$	$\boxed{102}$	103	102	101
Predetto[n-1]-1		99	100	101	102	$\boxed{103}$	$\boxed{102}$	$\boxed{101}$	$\boxed{100}$	99	100	$\boxed{101}$	$\boxed{100}$	$\boxed{99}$
Predetto[n]		101	102	103	104	103	102	101	100	101	102	101	100	99
Codifica		0	0	0	-1	0	0	0	0	-1	-1	0	0	-2