

# Fondamenti di Telecomunicazioni

Matteo Zanella

A.A 2022-2023

## 1 Segnali

Un segnale è una funzione del tempo  $s(t) \in \mathbb{R}$ , ad esempio una trasmissione su canale analogico/continuo avviene per mezzo di segnali. Il tempo può essere considerato come continuo o discreto:

1. I segnali a **tempo continuo** vengono indicati con  $y(t)$
2. I segnali a **tempo discreto** vengono indicati con  $s(KT)$  con  $T$  **quanto temporale** (tempo che intercorre tra due osservazioni sequenziali)

I valori assunti dai segnali possono essere:

1. **continui**: il segnale assume valori  $\in \mathbb{R}$  (o sottoinsiemi)
2. **discreti**: il segnale assume valori discreti

I segnali analogici sono segnali continui con valori continui, mentre i segnali digitali sono segnali discreti con valori discreti.

### 1.1 Esempi di segnali

$$rect(t) = \begin{cases} 0 & t < -\frac{1}{2} \vee t > \frac{1}{2} \\ 1 & t \in [-\frac{1}{2}, \frac{1}{2}] \end{cases}$$



$$\text{triang}(t) = \begin{cases} 0 & t < -1 \vee t > 1 \\ 1 & t \in [-1, 1] \end{cases}$$



## 1.2 Ritardo e scalamento di un segnale

Un **ritardo** di un segnale avviene quando l'intero segnale è trasposto di un tempo  $t_0$ :

$$s(t) = y(t - t_0)$$

**Amplificare o diminuire l'ampiezza** di un segnale si ottiene tramite:

$$s(t) = Ay(t)$$

**Dilatare o comprimere nel tempo** il segnale si ottiene tramite:

$$s(t) = y(At)$$

## 1.3 Campionamento di un segnale

Dato un segnale a tempo continuo  $s(t)$ , la sua versione campionata con un periodo di campionamento  $T$  ( $s(nT)$ ) trasforma il segnale da tempo continuo a tempo discreto a valori discreti.



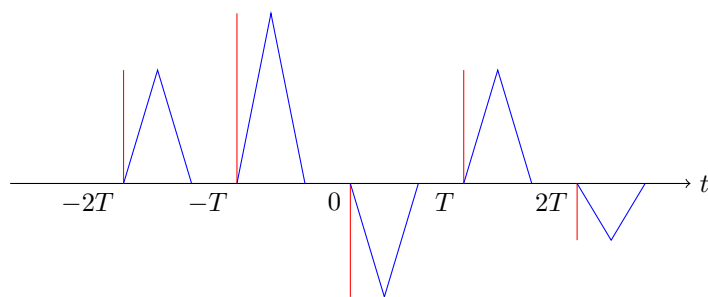
Utilizzare il campionamento provoca una perdita di qualità del segnale.

## 1.4 Interpolazione

Fisicamente non posso far avvenire variazioni immediate del segnale (far "esplodere" il segnale); l'interpolazione ci permette di passare da segnali a tempo discreto a un segnale a tempo continuo:

$$s(t) = \sum_{n=-\infty}^{\infty} y(nT)h(t - nT)$$

con  $h(t)$  risposta impulsiva dell'interpolatore.

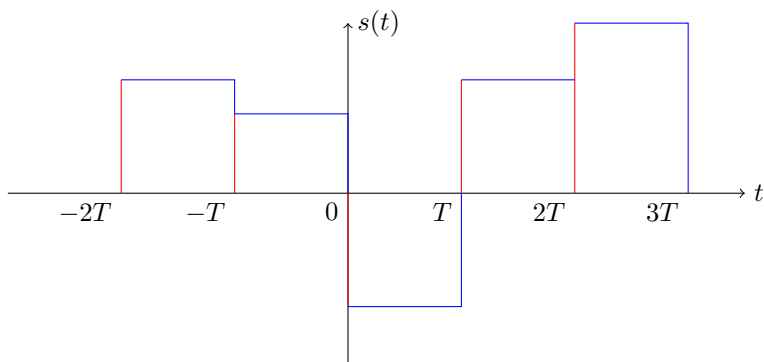


## 1.5 Holder

L'holder ci permette di prendere un segnale discreto composto da una serie di impulsi, e allungarne gli impulsi in modo da farli durare nel tempo:

$$h(t) = \text{rect}\left(\frac{t - \frac{T}{2}}{T}\right)$$

Applicando un holder a un segnale  $x(nT)$  si ottiene:



## 1.6 Energia di un segnale

Per un segnale a tempo continuo si ha che:

$$E_s = \int_{-\infty}^{\infty} s^2(t) dt$$

Per un segnale a tempo discreto, si usa la sommatoria  $E_s = \sum_{n=-\infty}^{\infty} s^2(nT)$

Per calcolare l'energia di un segnale attenuato abbiamo che:

$$E_{(As(t))} = A^2 E_s$$

## 1.7 Segnali in decibel

Per passare l'energia di un segnale in decibel si usa il seguente passaggio:

$$(E_s)_{dB} = 10 \log(E_s)$$

## 1.8 Spazio vettoriale di segnali a energia finita

### 1.8.1 Prodotto interno

Definiamo il prodotto interno tra due vettori come:

$$\langle x(t), y(t) \rangle = \int_{-\infty}^{\infty} x(t)y(t) dt$$

### 1.8.2 Norma di un vettore

Definiamo la norma di un vettore come:

$$\|s(t)\| = \sqrt{\langle s(t), s(t) \rangle} = \sqrt{E_s}$$

### 1.8.3 Base ortonormale

una base ortonormale di uno spazio vettoriale è un insieme di vettori  $\phi_1(t), \dots, \phi_I(t)$  tali che:

$$\langle x_i(t), x_j(t) \rangle = 0, \quad i \neq j, \quad \forall i, j$$

$$\|x_i(t)\| = 1, \quad \forall i$$

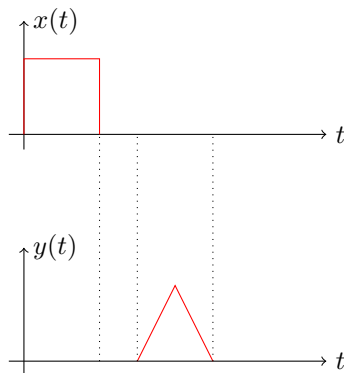
Deve inoltre valere che un segnale generico  $y(t)$  sia scrivibile come combinazione lineare dei vettori della base:

$$y(t) = \sum_{n=1}^I \langle y(t), x_n(t) \rangle x_n(t)$$

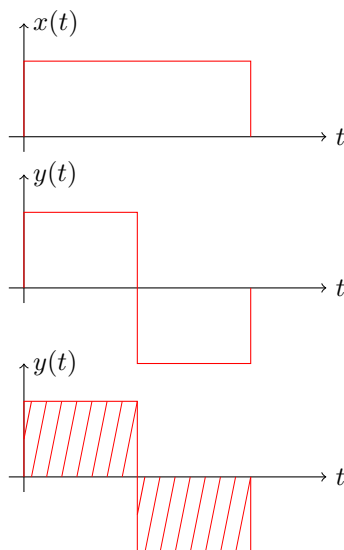
I vettori della base sono **ortogonali** in quanto il loro prodotto interno fa 0 e **normali** in quanto il modulo di ognuno di loro vale 1.

Da notare che due vettori sono ortogonali in due casi:

1. se i due segnali hanno **sostegni diversi**:



2. se l'area del prodotto dei due segnali è nulla



#### 1.8.4 Metodo di orto normalizzazione di Gram Schmidt

Supponiamo di avere  $N$  vettori  $s_1(t), \dots, s_N(t)$  e vogliamo trovare una base ortonormale per questi vettori: il primo passo è scegliere uno dei vettori e **normalizzarlo**, per usarlo come primo vettore della base:

$$\phi_1(t) = \frac{s_1(t)}{\sqrt{E_{s_1}}}$$

Ora dobbiamo trovare gli altri vettori della base con il seguente processo:

$$\phi'_i(t) = s_j(t) - \sum_{k=1}^{i-1} \langle s_k(t), \phi_i(t) \rangle \phi_k(t)$$

Ripetiamo quindi il processo finché non troviamo una base completa per i vettori dati.

## 2 Modulazione digitale

### 2.1 Ricevitore digitale

Il ricevitore digitale rappresenta un segnale ricevuto nello spazio della costellazione.

Un segnale trasmesso in un canale AWGN, sarà ricevuto nella seguente forma:

$$r(t) = s_{a_n}(t) + w(t)$$

$s_{a_n}(t)$  è il simbolo che era stato trasmesso, mentre  $w(t)$  è il **rumore che è stato introdotto dal canale**.

Il messaggio ricevuto è ovviamente una sequenza di simboli, i quali sono ricavabili applicando un ritardo di  $nT$  ai segnali.

### 2.2 Il rumore dopo la proiezione

Se trasmettiamo in un canale AWGN, il rumore è rappresentabile come una **v.a. Gaussiana** a media nulla e varianza indipendente dal rumore in altri istanti.

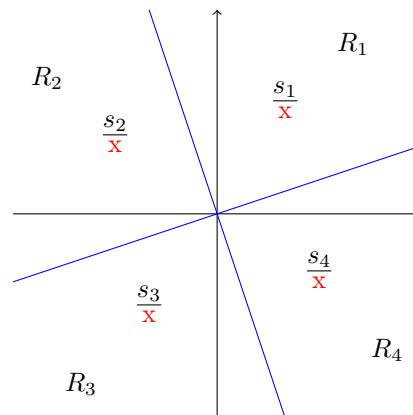
$$w(t) \sim N(0, \sigma_w^2)$$

Avendo media nulla, **l'energia del rumore si può approssimare a 0**.

### 2.3 Regioni di decisione

Per via del rumore, non sono certo di ricevere esattamente i simboli che vengono trasmessi, di conseguenza ho bisogno di determinare dei criteri per scegliere un segnale piuttosto che un altro, dato un segnale ricevuto.

Devo trovare un metodo per dividere lo spazio in  $M$  regioni, in modo da esaurire l'intero spazio euclideo, e non avere zone di indecisioni.



Per scegliere quale regione utilizzare, usiamo la **probabilità di decisione corretta**.

## 2.4 Probabilità di decisione corretta

Non sempre il ricevitore farà la decisione corretta, dobbiamo quindi calcolare la probabilità che il ricevitore la faccia:

$$P(C) = \sum_{n=1}^N \left( \int_{\mathfrak{R}_n} p_{r|a_0}(r|n) P_{a_0}(n) dr \right)$$

$N$  è il numero di regioni di decisione,  $\mathfrak{R}_n$  è la  $n$ -esima regione di decisione,  $p_{r|a_0}(r|n)$  è la densità di probabilità del rumore per un dato simbolo,  $P_{a_0}(n)$  è la probabilità di trasmissione del simbolo  $n$ . Il mio scopo è massimizzare la probabilità di decisione corretta: esistono vari criteri:

1. **MAP**:  $a_0 = \operatorname{argmax}_n \left( p_{a_0|n}(n|r) \right)$  da usare in caso generico
2. **ML**:  $a_0 = \operatorname{argmax}_n \left( p_{r|a_0}(r|n) \right)$  da usare quando i simboli trasmessi sono equiprobabili
3. **MD**:  $a_0 = \operatorname{argmin}_n \left( d^2(r, s_n) \right)$  da usare se si hanno simboli equiprobabili trasmessi in un canale AWGN

## 2.5 Ricevitori digitali

1. **MAP-ML**
2. **MD-1**
3. **MD-2**

## 2.6 Modulazione binaria

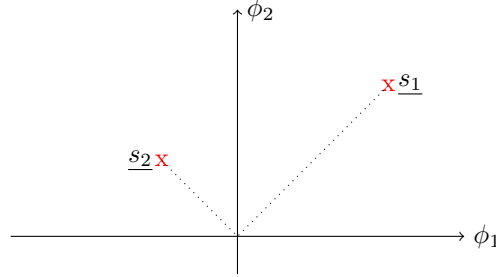
Nella modulazione binaria abbiamo **due simboli** ( $M = 2$ ) che mappano i valori di un bit. Sia  $I$  la dimensione della base:

1. **I=1**:





## 2. I=2:



$$|\underline{s}_i| = \sqrt{E_{s_i}}$$

Nel caso di una base a due vettori si ha che

$$\underline{s}_1 = \begin{bmatrix} \langle s_1(t), \phi_1(t) \rangle, \langle s_1(t), \phi_2(t) \rangle \end{bmatrix}$$

$$\underline{s}_2 = \begin{bmatrix} \langle s_2(t), \phi_1(t) \rangle, \langle s_2(t), \phi_2(t) \rangle \end{bmatrix}$$

### 2.6.1 Probabilità d'errore

$$P(E) = Q\left(\frac{d}{2\sigma_I}\right)$$

$Q(A)$  mi rappresenta l'area della gaussiana per i valori  $> A$

Per diminuire la probabilità d'errore bisogna aumentare la distanza tra i vettori o trovare un modo per diminuire la varianza del rumore.

### 2.6.2 Energia media della modulazione

$$E_s = E\left[E_{s_i}\right] = \sum_{i=1}^M E_{s_i} P_{a_0}(i)$$

L'energia media è il valore atteso delle energie dei simboli.

### 2.6.3 Coefficiente di correlazione

$$\rho = \frac{\langle \underline{s}_1, \underline{s}_2 \rangle}{\sqrt{E_{s_1} E_{s_2}}}$$

Se  $\rho = 0$  il prodotto interno è nullo, quindi i due segnali sono ortogonali.

Se  $\rho = -1$  è possibile (non è per forza vero) che i segnali siano antipodali; perché i segnali lo siano devono anche essere opposti ( $s_1(t) = -s_2(t)$ ).

Sfruttando il coefficiente di correlazione ci è possibile trovare le coordinate dei segnali anche senza conoscere la base della segnalazione:

$$\underline{s_1} = \begin{bmatrix} \sqrt{E_{s_1}}, 0 \end{bmatrix}$$

$$\underline{s_2} = \begin{bmatrix} \rho\sqrt{E_{s_2}}, \sqrt{E_{s_2}}\sqrt{1-\rho^2} \end{bmatrix}$$

## 2.7 Modulazioni M-arie

In generale, se abbiamo una segnalazione di **dimensione M**, valgono le seguenti regole:

### 2.7.1 Probabilità d'errore

1. **Limite inferiore:**

$$P(E) \geq \frac{1}{M} \sum_{m=1}^M Q\left(\frac{d_{min}}{2\sigma_I}\right)$$

2. **Limite superiore:**

$$P(E) \leq (M-1)Q\left(\frac{d_{min}}{2\sigma_I}\right)$$

3. **Modulazione ortogonale:**

$$\frac{M+1}{2}Q\left(\sqrt{\frac{E_s}{2\sigma_I^2}}\right) \leq P(E) \leq (M-1)Q\left(\sqrt{\frac{E_s}{2\sigma_I^2}}\right)$$

4. **Modulazione bi-ortogonale:**

$$\frac{M-1}{2}Q\left(\sqrt{\frac{E_s}{2\sigma_I^2}}\right) + \frac{1}{M}Q\left(\sqrt{\frac{E_s}{\sigma_I^2}}\right) \leq P(E) \leq (M-2)Q\left(\sqrt{\frac{E_s}{2\sigma_I^2}}\right) + Q\left(\sqrt{\frac{E_s}{\sigma_I^2}}\right)$$

## 2.8 Modulazione PAM

Nella modulazione PAM abbiamo un segnale base  $h_{TX}(t)$  usato come base, **scalato poi M volte**:

$$\begin{array}{ccccccc} \frac{s_1}{\text{X}} & \frac{s_2}{\text{X}} & \frac{s_3}{\text{X}} & \frac{s_4}{\text{X}} & & & \\ \hline -3\sqrt{E_h} & -\sqrt{E_h} & \sqrt{E_h} & 3\sqrt{E_h} & \rightarrow & \phi_1 & \end{array}$$

$$s_m(t) = \alpha_m h_{TX}(t) \quad \alpha_m = 2m-1-M, \quad m \in [1, M]$$

la base ha dimensione 1 e  $\phi(t) = \frac{h_{TX}(t)}{\sqrt{E_h}}$ .

La distanza tra due simboli adiacenti è di  $2\sqrt{E_h}$ .

### 2.8.1 Simboli

$$\underline{s}_m = \alpha_m \sqrt{E_h}$$

### 2.8.2 Energia media

$$E_s = \frac{M^2 - 1}{3} E_h$$

### 2.8.3 Probabilità d'errore

Analizzando una sezione per volta, notiamo due tipi di regioni di decisione: una esterna e una interna.

1. **Regione interna:**

$$P(E) = Q\left(\sqrt{\frac{E_h}{\sigma_I^2}}\right)$$

2. **Regione interna:**

$$P(E) = 2Q\left(\sqrt{\frac{E_h}{\sigma_I^2}}\right)$$

Andando a calcolare la probabilità d'errore totale abbiamo che, se i simboli sono trasmessi con la stessa probabilità:

$$P(E) = 2 \frac{M-1}{M} Q\left(\sqrt{\frac{E_s}{\sigma_I^2}}\right)$$

Utilizzando poi la mappatura di Gray per mappare i bit all'interno della segnalazione (due simboli diversi possono differire solo di un bit), otteniamo che la probabilità d'errore sul bit è di  $P_{bit} = \frac{P(E)}{\log_2(M)}$

## 2.9 Modulazione QAM

La modulazione QAM è simile alla PAM, solo che si sviluppa su una base di dimensione 2:



esempio di una 16-QAM

$$s_m(t) = \alpha_{m,I} h_{TX}(t) \cos(2\pi f_0 t + \varphi_0) - \alpha_{m,Q} h_{TX}(t) \sin(2\pi f_0 t + \varphi_0)$$

$$\alpha_{m,I}, \alpha_{m,Q} = 2l - L - 1 \quad m \in [1, M], \quad l = \sqrt{M}$$

### 2.9.1 Base

Abbiamo come base per la segnalazione:

$$\phi_1(t) = \sqrt{\frac{2}{E_{h_{TX}}}} \cos(2\pi f_0 t + \varphi_0) h_{TX}(t)$$

$$\phi_2(t) = -\sqrt{\frac{2}{E_{h_{TX}}}} \sin(2\pi f_0 t + \varphi_0) h_{TX}(t)$$

### 2.9.2 Simboli

$$\underline{s_m} = \left[ \sqrt{\frac{E_h}{2}} \alpha_{m,I}, \sqrt{\frac{E_h}{2}} \alpha_{m,Q} \right]$$

### 2.9.3 Energia media

$$E_s = \frac{M-1}{3} E_h$$

### 2.9.4 Probabilità d'errore



Analizzando una sezione per volta, notiamo 3 tipi di regioni di decisione: una interna, una sugli angoli e una sui lati.

1. **Regione sugli angoli:**

$$P(C|A) = \left(1 - Q\left(\sqrt{\frac{E_h}{2\sigma_I^2}}\right)\right)^2$$

2. **Regione sui lati:**

$$P(C|B) = \left(1 - 2Q\left(\sqrt{\frac{E_h}{2\sigma_I^2}}\right)\right)\left(1 - Q\left(\sqrt{\frac{E_h}{2\sigma_I^2}}\right)\right)$$

3. **Regione centrale:**

$$P(C|C) = \left(1 - 2Q\left(\sqrt{\frac{E_h}{2\sigma_I^2}}\right)\right)^2$$

Andando a calcolare la probabilità d'errore totale abbiamo che, se i simboli sono trasmessi con la stessa probabilità:

$$P(E) = 4\left(\frac{L-1}{L}\right)Q\left(\sqrt{\frac{3/2}{M-1} \frac{E_s}{\sigma_I^2}}\right)$$

Utilizzando poi la mappatura di Gray per mappare i bit all'interno della segnalazione (due simboli diversi possono differire solo di un bit), otteniamo che la probabilità d'errore sul bit è di  $P_{bit} = \frac{P(E)}{\log_2(M)}$

## 2.10 Quantizzazione

La quantizzazione è un processo che trasforma un segnale  $y(kT) \in \mathfrak{R}$  in  $y_q(kT) \in A$ , con  $A$  un alfabeto di valori discreti.

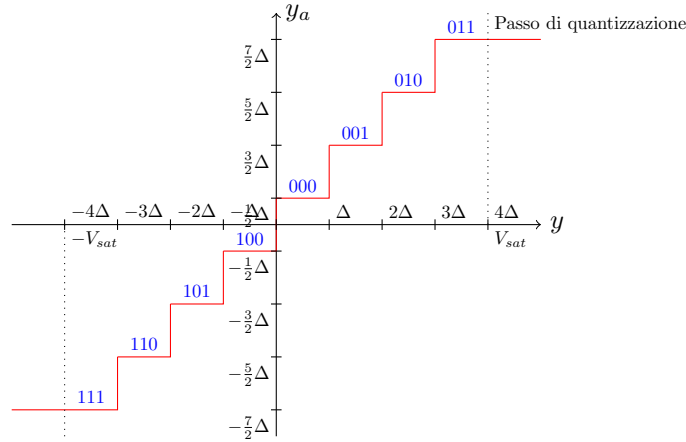
### 2.10.1 Funzione caratteristica

La funzione caratteristica di un quantizzatore indica i valori quantizzati in base al valore reale  $y(t)$ :



### 2.10.2 Quantizzatore uniforme

La funzione caratteristica di un quantizzatore uniforme è la seguente:



$$\Delta = \frac{2V_{sat}}{L}$$

$\Delta$  è detto passo di quantizzazione,  $L$  è il numero di livelli ( $L = 2^b$ ,  $b$  numero di bit).

In un quantizzatore possiamo trovare due tipi di errori:

1. **Errore granulare:** Quando  $y \in [-V_{sat}, V_{sat}]$

2. **Errore di saturazione:** Quando  $y \notin [-V_{sat}, V_{sat}]$

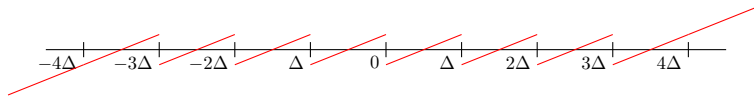
### 2.10.3 Gestione dell'errore di saturazione

Abbiamo due casi da distinguere:

1. Se il segnale è **limitato** in  $[-A, A]$  scelgo un  $V_{sat}$  in modo che  $V_{sat} = A$  per evitare un errore di saturazione
2. Se il segnale ha una **densità di probabilità con supporto non finito**, non posso impedire la saturazione;  
Scelgo una probabilità di saturazione che soddisfi le mie necessità, e calcolo  $V_{sat}$  di conseguenza:

$$P(y \notin [-V_{sat}, V_{sat}]) \leq P_{sat}$$

### 2.10.4 Gestione dell'errore granulare



$$e_q = y - y_a$$

L'errore granulare, ha valori all'interno dell'intervallo  $e_q \in [-\Delta/2; \Delta/2]$ . Supponendo di avere un numero  $L$  di livelli alto,  $\Delta$  assumerà un valore piccolo.

$$P_y(a) \simeq P_y(a_i) \quad a \in [a_i - \frac{\Delta}{2}, a_i + \frac{\Delta}{2}]$$

Supponendo che  $e_q$  sia uniforme in  $[-\Delta/2, \Delta/2]$ , si ha che il valore atteso dell'errore granulare è 0  $E[e_q] = 0$  mentre la varianza è  $\sigma_y = \frac{\Delta^2}{12}$

### 2.10.5 Rapporto segnale-rumore

$$(\Lambda)_{dB} = 20 \log\left(\frac{\sigma_y}{V_{sat}}\right) + 4.77 + 6.02b$$

Aumentare il numero di bit utilizzati, aumenta di 6 il SNR (Signal Noise Ratio).

### 3 Struttura dei sistemi di comunicazione

I sistemi di comunicazione coinvolgono **più tipi di dispositivi connessi**. Non c'è più una comunicazione punto punto, ma esiste una comunicazione tra più dispositivi.

Ci sono due concetti principali quando si parla di sistemi di comunicazione:

1. **Piano di controllo:** insieme delle procedure, tecniche e hardware necessario per il funzionamento di una rete
2. **Piano dei dati:** comunicazione dei dati dell'utente

#### 3.1 Protocolli di comunicazione

Un protocollo di comunicazione è un insieme di regole per lo scambio dei messaggi tra più dispositivi.

##### 3.1.1 Standard OSI

Da' un quadro generale per creare uno standard di comunicazione. Lo **standard OSI** separa uno standard di telecomunicazione in 7 strati. Ogni strato di un dispositivo A deve comunicare con il rispettivo strato di un dispositivo B, attraversando tutti gli strati inferiori.

1. **Fisico:** Si occupa della trasmissione dei dati su uno specifico mezzo di comunicazione
2. **Collegamento dati:** Creare una comunicazione affidabile e coordinare la trasmissione tra più dispositivi (migliorare e tenere sotto controllo gli errori che possono avvenire nello strato fisico)
3. **Rete:** Si occupa dell'instradamento: non siamo più in situazione di punto punto, ma serve inviare messaggi attraverso dispositivi diversi prima di raggiungere la destinazione finale
4. **Trasporto:** Rende affidabile il collegamento end-to-end nella rete
5. **Sessione:** Stabilisce il periodo nel quale due entità di parlano tra loro; Si stabiliscono delle regole (es attivare e disattivare canali di comunicazione per parlare a turni)
6. **Presentazione:** Troviamo ciò che riguarda il contenuto del messaggio
7. **Applicazione:** Gestione completa dell'applicazione/del servizio



## 3.2 Codifica di canale

Tecniche che ci permettono di **rendere affidabile** la comunicazione e correggere (in parte) gli errori sul bit.

Ci sono due strategie generali:

1. **Ritrasmissione**

chiediamo al mittente di **ripetere** un messaggio;  
dobbiamo essere in grado di capire se c'è stato un errore e dobbiamo essere in grado di poter comunicare al mittente la richiesta di ritrasmissione

2. **Codifica del messaggio in trasmissione**

elaboro la sequenza di bit (aggiungo dei bit) calcolati sui bit del messaggio, che mi aiutano di **correggere alcuni errori** (es bit di parità);  
Questa tecnica mi allunga il pacchetto di bit, e c'è una possibilità di non dover utilizzare questi bit (nel caso non ci siano stati errori)

Ci sarebbe anche una terza strategia ibrida: una ritrasmissione in cui vengono mandati solo dei bit di codifica.

## 3.3 Canale Binario

Abbiamo un canale binario che trasmette dei bit  $c_k \in \{0, 1\}$ ; il ricevitore tuttavia riceve dei bit  $\hat{c}_k \in \{0, 1\}$  non per forza uguali ai bit trasmessi.

Il canale binario può essere rappresentato come una serie di un modulatore digitale, un canale (es AWGN) e un demodulatore digitale.

Nel caso in cui la probabilità di sbagliare un bit invece che un altro, si dice che ci troviamo in un **canale binario simmetrico senza memoria**.

$$P(\hat{c}_k = 1 | c_k = 0) = P(\hat{c}_k = 0 | c_k = 1) = P_{bit}$$

Inoltre, in un canale del genere, gli errori su bit distinti sono indipendenti.

Attenzione che non tutti i canali binaria si comportano in questo modo, è una situazione ideale ma che ci semplificherà i calcoli e le analisi dei canali.

Il modello a cui facciamo riferimento per correggere gli errori è quello dei **codici di canale a blocco**.

### 3.3.1 Codici di canale a blocco

A partire da blocchi di  $k$  bit, generiamo blocchi di  $n$  bit, con  $n > k$  (sia  $k$  che  $n$  saranno nomi di variabili riservate a queste nozioni).

I blocchi di ingresso (lungi  $k$ ) contengono il messaggio che vogliamo trasmettere, quelli di uscita includono anche un'informazione per correggere gli errori.

$$\underline{b} = [b_1, \dots, b_k] \rightarrow \underline{c} = [c_1, \dots, c_n] \rightarrow \text{CAN BIN} \rightarrow \hat{\underline{c}} = [\hat{c}_1, \dots, \hat{c}_n] \rightarrow \hat{\underline{b}} = [\hat{b}_1, \dots, \hat{b}_k]$$

I possibili vettori  $\underline{b}$  in ingresso sono  $2^k$ ; i possibili messaggi  $\underline{c}$  che escono dal codificatore sono comunque  $2^k$ , in quanto il numero di messaggi codificati è pari al numero di messaggi da codificare, i quali sono  $2^k$ .

Al ricevitore invece, per via dei disturbi all'interno del canale, ricevo  $2^n$  messaggi  $\underline{\tilde{c}}$  da decodificare, i quali diventeranno di nuovo  $2^k$  messaggi decodificati  $\underline{\hat{b}}$ .

L'insieme dei possibili vettori  $\underline{c}$  lo chiamiamo **codice**  $\mathcal{C}$ , mentre i vettori  $\underline{c}$  li chiameremo **parole di codice**.

I vettori  $\underline{\tilde{c}}$  non faranno sempre parte del codice, quindi non sono, in generale, parole del codice ma semplici sequenze binarie.

I bit della sequenza binaria  $\underline{\tilde{c}}$  possono essere scritti come  $\tilde{c}_i = c_i + e_i$ , con  $e_i$  **errore introdotto dal canale** (somma binaria senza riporto,  $1 + 1 = 0$ ), se  $e_i = 0$  non ho un errore sul bit  $i$ , altrimenti ho un errore.

In un canale binario simmetrico senza memoria, gli  $e_i$  sono indipendenti e definiamo  $P_{bit} = P(e_i = 1) \quad \forall i$ .

### 3.4 Criteri di decodifica

Come passare da  $\underline{\hat{c}}$  a  $\underline{\hat{b}}$ :

Volendo minimizzare la probabilità di errore sulle sequenza,  $P_e = (P(\underline{\hat{b}} \neq \underline{b}))$

#### 1. Criterio MAP

$$\underline{\hat{c}} = \underset{\underline{\alpha} \in \mathcal{C}}{\operatorname{argmax}} (P(\underline{\hat{c}} = \underline{\alpha} | \underline{c} = \underline{\alpha}) P(\underline{c} = \underline{\alpha}))$$

#### 2. Criterio di massima verosimiglianza (ML)

$$\underline{\hat{c}} = \underset{\underline{\alpha} \in \mathcal{C}}{\operatorname{argmax}} (P(\underline{c} = \underline{\alpha} | \underline{\hat{c}} = \underline{\beta}))$$

Da usare in caso di parole di codice equiprobabili.

#### 3. Criterio a minima distanza

Usando la **distanza di Hamming**, che determina la distanza tra due parole in base al numero di bit per cui differiscono, calcolo che

$$\underline{\hat{c}} = \underset{\underline{\alpha} \in \mathcal{C}}{\operatorname{argmin}} (d_H(\underline{\alpha}, \underline{\hat{c}}))$$

### 3.5 Errori rilevati e corretti da un codice a blocco

Il numero massimo di **errori che posso correggere** è  $\lfloor \frac{d_{H,min}-1}{2} \rfloor$ .

Il numero massimo di **errori che posso rilevare** è invece di  $d_{H,min} - 1$ .

Con  $d_{H,min}$  la distanza di Hamming minima tra due parole di codice.

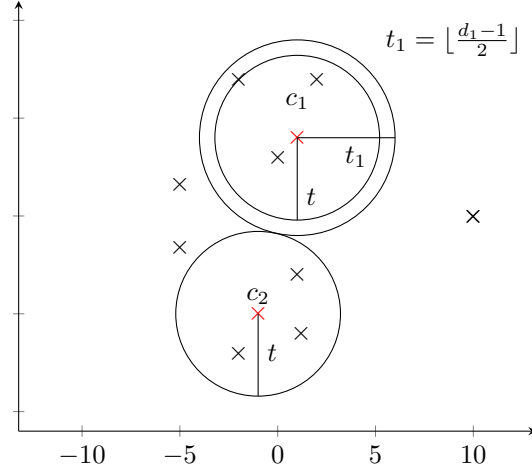
#### 3.5.1 Distanza di Hamming minima di un codice $\mathcal{C}$

$$d_{min} = \min(d_H(\underline{C}_i, \underline{C}_j))$$

Per calcolare queste distanze di Hamming dovremmo analizzare tutta la tabella; possiamo piuttosto calcolare la relazione tra  $d_{min}$  e  $k$  numero di bit nella parola di codice.

### 3.5.2 Bound di Hamming

Le sequenze che si trovano all'interno di una distanza di  $\lfloor \frac{d_{min}-1}{2} \rfloor$  sono sequenze che riesco sempre ad associare ad una parola di codice, quindi a correggerle. Gli errori al di fuori da questa distanza non sempre riesco a correggerli...



Le sequenze che trovo all'interno di questi bound di Hamming sono rappresentabili come  $\underline{c}_i + \underline{e}$ , in cui ogni vettore  $\underline{e}$  rappresenta l'errore introdotto dal canale, e ha una quantità di bit 1 minore o uguale al valore  $t_i$  del bound di Hamming.

Il numero di sequenze all'interno di ogni bound di Hamming è pari a

$$\sum_{r=0}^{t_i} \binom{n}{r}$$

Prendendo  $t$  la distanza minima tra le parole di codice  $t = \lfloor \frac{d_{min}-1}{2} \rfloor$ , le sequenze che hanno distanza fino a  $t$  dalle parole di codice, e che quindi posso correggere, sono

$$2^k \sum_{r=0}^t \binom{n}{r}$$

Il numero di sequenze totali che posso avere è di  $2^n$ , quindi abbiamo che

$$2^k \sum_{r=0}^t \binom{n}{r} \leq 2^n$$

Sviluppando i calcoli, arrivo ad ottenere che

$$\frac{k}{n} \leq 1 - \frac{1}{n} \log_2 \left( \sum_{r=0}^t \binom{n}{r} \right)$$

conosciuto come **bound di Hamming**.

Dalla formula ricavata scopro che, aumentando  $t$  (ovvero aumentando il numero di errori che voglio correggere), diminuisco il rapporto  $\frac{k}{n}$

### 3.6 Codici lineari a blocco

Un vettore binario lungo  $n$  è una sequenza di  $n$  bit.

Definisco l'operazione di somma tra due vettori come la somma binaria senza riporto:

$$\underline{x} + \underline{y} = \underline{z}$$

L'insieme degli scalari  $\alpha$  è ovviamente  $\{0, 1\}$ :

$$\alpha \underline{x} = \begin{cases} \underline{x} & \alpha = 1 \\ \underline{0} & \alpha = 0 \end{cases}$$

#### 3.6.1 Norma

Definiamo come **norma di un vettore** (o **peso di Hamming**)  $||\underline{x}||_H$ , il numero di 1 all'interno del vettore.

Abbiamo alcune regole per la norma di un vettore:

1.

$$||\underline{x}||_H \geq 0$$

2.

$$||\underline{x}||_H = 0 \iff \underline{x} = \underline{0}$$

3.

$$||\alpha \cdot \underline{x}||_H = \alpha ||\underline{x}||_H$$

4.

$$||\underline{x} + \underline{y}||_H = ||\underline{x}||_H + ||\underline{y}||_H$$

La distanza indotta dal peso di Hamming è anche la distanza di Hamming:

$$||\underline{x} - \underline{y}||_H = d_H(\underline{x}, \underline{y})$$

Attenzione che essendo in caso di somme binarie senza riporto, vale la seguente relazione:

$$\underline{x} + \underline{y} = \underline{x} - \underline{y}$$

$\mathcal{C}$  è un **codice lineare a blocco** se  $\{\underline{c}_1, \dots, \underline{c}_{2^k}\}$  è un sottospazio lineare dello spazio binario di dimensione  $n$ :  $\mathcal{C}$  è lineare  $\iff \alpha \underline{c}_1 + \beta \underline{c}_2 \in \mathcal{C} \quad \forall \underline{c}_1, \underline{c}_2$

### 3.6.2 Peso di Hamming di un codice

Il **peso di Hamming di un codice** è il peso di Hamming minimo tra tutte le parole di codice non nulle

$$W_H(\mathcal{C}) = \min \|\underline{c}\|_H \quad \underline{c} \in \mathcal{C} \setminus \{0\}$$

In un codice lineare a blocco, il peso di Hamming del codice, coincide con la distanza minima del codice

$$W_H(\mathcal{C}) = \min(d_H(\underline{\gamma}_1, \underline{\gamma}_2))$$

In quanto il codice è lineare, la distanza di Hamming tra due parole di codice, ovvero la differenza tra i due vettori, è il peso di Hamming di una parola di codice:  $d_H(\underline{\gamma}_1, \underline{\gamma}_2) = \|\underline{c}\|_H$ ,  $\underline{c} \in \mathcal{C}$ , quindi abbiamo che

$$W_H(\mathcal{C}) = \min(d_H(\underline{\gamma}_1, \underline{\gamma}_2)) = \min \|\underline{c}\|_H, \quad \underline{c} \in \mathcal{C} \setminus \{0\}$$

### 3.6.3 Matrice generatrice di un codice lineare

Una matrice  $\underline{G}$  è matrice generatrice del codice  $\mathcal{C}$  se la generica parola di codice può essere scritta come

$$\underline{c} = \underline{G} \underline{b}$$

$$\underline{c} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix} \quad \underline{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_k \end{pmatrix}$$

$$\underline{G} = \begin{pmatrix} G_{1,1} & \cdot & G_{1,k} \\ \vdots & \cdot & \vdots \\ G_{n,1} & \cdot & G_{n,k} \end{pmatrix}$$

per ogni vettore  $\underline{b}$

Le colonne di  $\underline{G}$  sono **linearmente indipendenti**, altrimenti non potrei avere  $2^k$  diverse parole di codice.

Cambiare l'ordine delle righe di  $\underline{G}$  cambia l'ordine dei bit in ogni parola di codice, ottenendo un codice diverso: usando questo codice, dato che il peso di Hamming non viene modificato, la distanza tra le parole di codice non cambia, mantenendo le stesse prestazioni e probabilità d'errore del codice originale.

Da una qualsiasi matrice generatrice posso sempre ottenere una matrice generatrice  $\underline{G}'$  di un codice equivalente in forma sistematica utilizzando combinazioni lineari delle colonne o permutazioni delle righe.

Con matrice  $\underline{G}$  sistematica:

$$\underline{G} = \left( \begin{array}{c} \underline{I} \\ \hline \underline{A} \end{array} \right) \quad \underline{I}[k \times k], \underline{A}[(n-k) \times k]$$

generando le parole di codice

$$\underline{c} = \left( \begin{array}{c} \underline{b} \\ \hline \underline{A} \underline{b} \end{array} \right)$$

### 3.6.4 Bound di Singleton

Per codici lineari a blocco  $(n, k)$ , la distanza minima del codice è limitata superiormente

$$d_{min} \leq n - k + 1$$

in quanto una matrice  $\underline{G}$  in forma sistematica avrà come massimo  $1 + n - k$  uni per colonna.

$$\underline{G} = \left( \begin{array}{c} \underline{I} \\ \hline \underline{A} \end{array} \right) \quad \underline{I}[k \times k], \underline{A}[(n-k) \times k]$$

Come conseguenza di questa conclusione c'è il fatto che non possiamo quindi distanziare troppo le parole di codice tra loro, impedendoci di abbassare troppo la probabilità d'errore.

## 3.7 Decodifica dei codici lineari

Una matrice di controllo di parità di un codice lineare  $\mathcal{C}$  soddisfa

$$\underline{H}\underline{\gamma} = \underline{0} \iff \underline{\gamma} \in \mathcal{C}$$

tecnica utile per determinare se un vettore  $\underline{\gamma}$  è una parola del codice.

$\underline{H}$  è matrice di controllo di parità di un codice con matrice generatrice  $\underline{G}$  se e solo se

$$\underline{H}\underline{G} = \underline{0} \quad e \quad rango(\underline{H}) = n - k$$

### 3.7.1 Codice con Matrice $\underline{G}$ in forma sistematica

$$\underline{G} = \left( \begin{array}{c} \underline{I} \\ \hline \underline{A} \end{array} \right) \quad \underline{I}[k \times k], \underline{A}[(n-k) \times k]$$

Consideriamo la matrice di parità  $\underline{H}$  per questa matrice  $\underline{G}$ :

$$\underline{G} = \left( \begin{array}{cc} \underline{A} & \underline{I} \end{array} \right) \quad \underline{A}[(n-k) \times k], \underline{I}[(n-k) \times (n-k)]$$

### 3.7.2 Decodifica per codici lineari

Siamo in un canale binario simmetrico senza memoria (si usa il criterio a minima distanza).

Se riceviamo la sequenza  $\underline{a} = (a_1, \dots, a_n)$ , la parola decodificata è

$$\hat{\underline{c}} = \operatorname{argmin}_{\underline{c} \in \mathcal{C}} \|\underline{c} - \underline{a}\|_H$$

Definiamo  $\underline{\epsilon} = \underline{c} - \underline{a}$ .

$$\hat{\underline{c}} = \underline{a} + \operatorname{argmin}_{\underline{\epsilon}} \|\underline{\epsilon}\|$$

### 3.7.3 Sindrome di una sequenza a

$$\underline{\sigma} = \underline{H}\underline{a}$$

Se  $\underline{a}$  è una parola di codice,  $\underline{\sigma} = 0$ .

Sappiamo tuttavia che  $\underline{a} = \underline{c} + \underline{\epsilon}$ , quindi  $\underline{a} \notin \mathcal{C}$

$$\underline{H}\underline{a} = \underline{H}(\underline{\epsilon} + \underline{c}) = \underline{H}\underline{\epsilon}$$

Da questa equazione ricaviamo che tutte le sequenze che hanno lo stesso errore  $\underline{\epsilon}$  hanno la stessa sindrome.

$$\hat{\underline{c}} = \underline{a} + \underline{\epsilon}_{\min}(\underline{\sigma})$$

## 3.8 Codici di Hamming

Un codice di Hamming è un codice lineare a blocco che ha come matrice di Parità  $\underline{H}$  che contiene come colonne tutte le sequenze non nulle di lunghezza  $n - k$ .

## 4 Teoria dell'informazione

Facciamo riferimento a uno spazio degli eventi  $\Omega$ , mentre sottoinsiemi di  $\Omega$  rappresentano eventi distinti  $A \subseteq \Omega$ .

Sullo spazio degli eventi, definiamo la probabilità  $P$

### 4.1 Assiomi dell'informazione

L'informazione è una funzione dell'evento  $A$  ( $i(A)$  importanza/quantità dell'informazione) in cui vale:

1.  $i(A) \geq 0$ ,  $\forall A$   
L'informazione deve esserci per esistere.
2.  $P(A) \leq P(B)$ ,  $i(A) \geq i(B)$   
Se un'informazione è **meno probabile**, allora sarà **più importante**.
3.  $i(\Omega) = 0$   
Siccome  $\Omega$  è lo spazio degli eventi, la probabilità che accada  $\Omega$  è 1, quindi **non è importante dato che accade sicuramente**.
4. Se  $A$  e  $B$  sono eventi indipendenti allora  $i(\{A \wedge B\}) = i(A) + i(B)$  (ricordiamo che  $P(A \wedge B) = P(A)P(B)$ )

Andando ad analizzare matematicamente l'informazione relativa ad un evento proviamo a spiegarla tramite la relazione

$$i(A) = \pm \log_-(P(A))$$

Andiamo a verificare se gli assiomi valgono:

1. Siccome la probabilità è compresa tra 0 e 1, il logaritmo ha risultati negativi: dobbiamo usare il meno
2.  $-\log$  è una funzione decrescente, quindi all'aumentare della probabilità, diminuisce l'informazione
3.  $P(\Omega) = 1 \rightarrow \log(1) = 0$
4.  $i(A + B) = -\log(P(A \wedge B)) = -(\log(P(A)P(B))) = -(\log(P(A)) + \log(P(B))) = i(A) + i(B)$

La relazione trovata quindi diventa:

$$i(A) = -\log_b(P(A)), \quad b > 1$$

Usando la base 2, l'informazione sarà **rappresentabile in bit**.

Usando la base  $e$ , l'unità di misura diventa il **Neper**.



## 4.2 Funzione informazione di una variabile aleatoria discreta

Sia  $X$  una variabile aleatoria discreta con densità di probabilità  $p_x(x)$ :

$$x \in A_x = \{a_1, \dots, a_n\}$$

$$p_x(a) = P(X = a), \quad a \in A_x$$

Definiamo la **funzione informazione** come

$$i_x(a) = i(\{x = a\}) = -\log_2(p_x(a))$$

### 4.2.1 Informazione di eventi di $X$

$$x \in A_x = \{a_1, \dots, a_n\}$$

Dato un insieme  $B \subseteq A_x$ ,

$$i(B) = l \log_2(P(B)) = -\log_2\left(\sum_{a \in B} p_x(a)\right)$$

Noto quindi che la funzione informazione non mi serve per calcolare l'informazione di un evento specifico, devo per forza passare per la densità di probabilità.

## 4.3 Entropia di una variabile aleatoria

Data una v.a. discreta  $X$ , l'entropia di  $X$  è definita come l'**informazione media** di  $X$ :

$$H(x) = E[i_x(x)] = \sum_{a \in A_x} p_x(a) \log_2\left(\frac{1}{p_x(a)}\right)$$

L'entropia mi rappresenta l'informazione media portata dall'informazione, quindi la novità portata dalla realizzazione di un determinato evento.

Un altro modo per vedere l'entropia è che l'entropia mi rappresenta l'incertezza complessiva sulla variabile aleatoria.

### 4.3.1 Proprietà dell'entropia

1.  $H(X) \geq 0$
2.  $H(X) = 0$  solo se  $X$  è deterministica

## 4.4 Funzioni concave e disuguaglianza di Jensen

Una funzione  $f(x)$  è strettamente concava, dati  $k_1, \dots, k_N \geq 0$ , tali che  $\sum_{n=1}^N k_n = 1$  e dati  $a_1, \dots, a_N$  distinti, se vale che

$$\sum_{n=1}^N k_n f(a_n) < f\left(\sum_{n=1}^N (k_n a_n)\right)$$

#### 4.4.1 Disequazione di Jensen

Sia  $X$  v.a. e  $f(x)$  una funzione strettamente concava:

$$E[f(x)] = \sum_{a \in A_x} p_x(a) f(a)$$

$$E[f(x)] < f\left(\sum_{a \in A_x} p_x(a) a\right) = f(E[x])$$

Si ha quindi che

$$E[f(x)] < f(E[x])$$

#### 4.5 Limite superiore dell'entropia di una v.a. discreta

Sia  $X$  una v.a. discreta  $X \in A_x$ ; indichiamo con  $M$  la cardinalità dell'alfabeto  $M = |A_x|$ .

Il limite superiore dell'entropia di  $X$  è

$$H(x) \leq \log_2(M)$$

inoltre  $H(x) = \log_2(M) \iff X$  è uniforme in  $A_x$ .

Dim:

$$H(x) = E[-\log_2(p_x(x))]$$

Ricordando che  $\log_2(a)$  è strettamente concava, sfruttiamo la disequazione di Jensen:

$$H(x) = E[\log_2(\frac{1}{p_x(x)})] < \log_2\left(E[\frac{1}{p_x(x)}]\right) = \log_2\left(\sum_{a \in A_x} \frac{p_x(a)}{p_x(a)}\right) = \log_2(M)$$

La disequazione vale solo se i valori di  $p_x(x)$  sono distinti tra loro.

Se  $X$  è uniforme nell'alfabeto  $(p_x(a) = \frac{1}{M})$ .

$$H(X) = E[\log_2(\frac{1}{p_x(x)})] = \sum_{a \in A_x} \frac{1}{M} \log_2(\frac{1}{\frac{1}{M}}) = \frac{1}{M} \sum_{m=1}^M \log_2(M) = \log_2(M)$$

#### 4.6 Vettori aleatori

Un vettore aleatorio di lunghezza  $N$   $\underline{X} = [X_1, \dots, X_N]$ ,  $n \in [1, N]$  con  $X_n$  variabile aleatoria discreta con alfabeto  $A_n$ ,  $\underline{X}_n \in A_1 \times \dots \times A_N = A$ .

La densità di probabilità del vettore aleatorio  $p_{\underline{x}}(\underline{a}) = P(\underline{X} = \underline{a})$ ,  $\underline{a} \in A$

#### 4.7 Funzione informazione per un vettore aleatorio

Definito quindi il vettore aleatorio, abbiamo che la funzione informazione, per un vettore aleatorio, è:

$$i_{\underline{x}}(\underline{a}) = \log_2\left(\frac{1}{p_{\underline{x}}(\underline{a})}\right)$$

## 4.8 Entropia di una variabile aleatoria

L'entropia di un vettore aleatorio, come per una variabile aleatoria discreta, è

$$H(X) = E[i_x(X)] = \sum_{a \in A} p_x(a) \log_2 \left( \frac{1}{p_x(a)} \right)$$

## 4.9 Entropia di più variabili aleatorie

Consideriamo due v.a.  $X$   $Y$ .

1. Se  $X$  e  $Y$  sono **indipendenti**:

$$H(X, Y) = E[i_{x,y}(X, Y)] = E[i_x(a) + i_y(b)] = H(X) + H(Y)$$

2. Se  $X$  e  $Y$  **non sono indipendenti**:

$$H(X, Y) < H(X) + H(Y)$$

3. Se  $f(X) = Y$  è **funzione deterministica**:

$$H(X, Y) = H(X)$$

La cosa tuttavia non ci sorprende in quanto l'entropia esprime l'incertezza sulla coppia, e essendo  $Y$  funzione deterministica di  $X$ , l'incertezza della coppia è quella di  $X$ .

Attenzione che non vale in generale il contrario, ovvero

$$H(X, Y) \neq H(Y)$$

4. Se  $Y$  **non è funzione deterministica** di  $X$ :

$$H(X, Y) > H(X)$$

5. Se  $f(X) = Y$  è **funzione iniettiva**, la densità di probabilità di  $X$  è la stessa di  $Y$  a meno di una modifica dell'alfabeto:

$$H(X) = H(Y) = H(X, Y)$$

Volendo raccogliere tutte le disuguaglianze ricavate, abbiamo la seguente catena:

$$0 \leq \max\{H(X, Y)\} \leq H(X, Y) \leq H(X + Y)$$

Vale sempre che  $H(X, Y) = H(Y, X)$ .

#### 4.9.1 Variabili aleatorie condizionate

Data la v.a.  $X$  condizionata alla variabile aleatoria  $Y$  con densità di probabilità condizionata  $p_{x|y}(a|b) = P(X = a|Y = b)$ , definiamo la funzione informazione condizionata come

$$i_{x|y}(a|b) = \log_2 \left( \frac{1}{p_{x|y}(a|b)} \right)$$

**Entropia condizionata ad un certo valore di  $y$ :**

$$H(X|Y = b) = \sum_{a \in A_x} p_{x|y}(a|b) \log_2 \left( \frac{1}{p_{x|y}(a|b)} \right)$$

In generale

$$H(X|Y) = E_y[H(X|Y = y)] = \sum_b p_y(b) H(X|Y = b)$$

Sviluppando i calcoli nella sommatoria, otteniamo un tipo di entropia diversa dal solito:

$$H(X|Y) = \sum_{a,b} p_{x,y}(a,b) \log_2 \left( \frac{1}{p_{x|y}(a|b)} \right)$$

Il nostro scopo è quello di scrivere l'entropia condizionata in funzione dell'entropia congiunta... sviluppando i calcoli nella sommatoria, si ottiene:

$$H(X, Y) = H(X|Y) + H(Y)$$

Questa conclusione si poteva ricavare anche dalla relazione tra probabilità congiunta e probabilità condizionata, in quanto l'entropia è una funzione logaritmica e trasforma i prodotti in somme.

#### 4.10 Informazione mutua tra due variabili aleatorie

$$I(X; Y) = I(Y; X) = H(X) + H(Y) - H(X, Y)$$

Dalla relazione trovata tra entropia condizionata e congiunta possiamo semplificare la formula dell'informazione mutua:

$$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

A livello logico, sto togliendo dall'informazione che mi porta  $X$ , l'informazione su  $X$  che già sapevo data l'informazione  $Y$ .

$H(X|Y)$  misura **quanto non so di  $x$  dopo che ho visto  $y$** .

$I(X; Y)$  misura **quanto so di  $x$  dopo aver visto  $y$** .

#### 4.10.1 Valori assunti dalla mutua informazione

$$I(X; Y) = H(X) + H(Y) - H(X, Y) \geq 0$$

In quanto sappiamo che l'entropia congiunta vale meno della somma delle entropie di  $X$  e  $Y$  (è impossibile avere un'informazione nulla, l'informazione o esiste ( $>0$ ) o non esiste ( $=0$ )).

$$I(X; Y) = H(X) - H(X|Y) \leq \min\{H(X), H(Y)\}$$

Si ha quindi che

$$0 \leq I(X; Y) \leq \min\{H(X), H(Y)\}$$

#### 4.11 Entropia per simbolo di un vettore aleatorio

Dato un vettore aleatorio  $\underline{X} = [X_1, \dots, X_N]$ , l'entropia per simbolo del vettore è

$$H_s(\underline{X}) = \frac{1}{N} H(\underline{X})$$

Se gli elementi del vettore aleatorio  $\underline{X}$  sono **identicamente distribuiti** ( $p_{X_i}(a) = p_{X_j}(a)$ ,  $\forall i$ ), abbiamo che l'entropia per simbolo coincide con l'entropia del vettore aleatorio

$$H_s(\underline{X}) = \frac{1}{N} H(\underline{X}) = H(X)$$

dato che, in quanto identicamente distribuiti,  $H(\underline{X}) = NH(X)$ .

#### 4.12 Informazione mutua per simbolo

$$I_s(\underline{X}; \underline{Y}) = H_s(\underline{X}) + H_s(\underline{Y}) - H_s(\underline{X}, \underline{Y})$$

#### 4.13 Sorgente di simboli

Se abbiamo una sorgente che in un tempo infinito genera un numero infinito di simboli, abbiamo un vettore aleatorio  $\underline{X} = [\dots, X_{-1}, X_0, X_1, \dots]$  di dimensione infinita.

L'entropia di solito è infinita in tutto il vettore, se questo è infinitamente lungo; l'entropia per simbolo invece sarà

$$H_s(\underline{X}) = \lim_{N \rightarrow \infty} \frac{1}{2N+1} H(\underline{X} = [X_{-N}, \dots, X_N])$$

#### 4.14 Tasso di informazione di un messaggio

Una sorgente genera simboli  $X_i$  ad intervalli di tempo  $T$  = periodo di simbolo. Il tasso di informazione (information rate) del messaggio generato dalla sorgente è

$$R(\underline{X}) = \frac{H_s(\underline{X})}{T} \quad [bit/s]$$

## 4.15 Tasso nominale del messaggio

Il tasso nominale del messaggio con  $X_i \in A$  con  $A$  di cardinalità  $M$  è

$$\frac{\log_2(M)}{T}$$

Il tasso di informazione di un messaggio mi dice **quanta informazione viene generata dalla sorgente al secondo**, mentre il tasso nominale del messaggio mi dice semplicemente **quanti bit vengono trasmessi** (non per forza contenenti informazione  $i = 0$ ).

Ovviamente il tasso nominale del messaggio fungerà come limite superiore del tasso di informazione di un messaggio, in quanto non può essere trasmessa più informazione di quanti bit vengono trasmessi

$$R(\underline{X}) \leq \frac{\log_2(M)}{T}$$

## 5 Codifica di canale

Indichiamo con  $T_c$  il **tempo di simbolo del canale**.

### 5.1 Canale numerico M-ario senza memoria

Essendo un canale numerico,  $x_n, y_n \in \{a_1, \dots, a_M\}$ , ed essendo senza memoria,  $y_n$  dipende solo da  $x_n$  e non dai simboli trasmessi precedentemente.

#### 5.1.1 Descrizione del canale attraverso una matrice di probabilità di transizione

$$P(y_n = a | x_n = b) = \begin{pmatrix} P(1,1) & \cdot & P(1,M) \\ \cdot & \cdot & \cdot \\ P(M,1) & \cdot & P(M,M) \end{pmatrix}$$

Sommando i valori della colonna  $i$  della matrice, a patto che ogni simbolo sia trasmesso con la stessa probabilità, ottengo  $P_{y_n}(i)$ ; sommando le colonne della matrice, invece, ottengo 1.

#### 5.1.2 Tasso di informazione del canale

Definiamo  $\mathcal{G}$  un canale numerico M-ario senza memoria, con probabilità di transizione  $P(y_n | x_n)$ .

Il tasso di informazione di  $\mathcal{G}$  è (per una certa statistica della sorgente)

$$R(\mathcal{G}) = \frac{1}{T_c} I_s(\underline{x}; \underline{y}) = \frac{1}{T_c} [H_s(\underline{x}) + H_s(\underline{y}) - H_s(\underline{x}, \underline{y})] \quad [bit/s]$$

Se  $x_n$  e  $y_n$  sono **indipendenti** ((in  $n$ ) e **identicamente distribuiti**:

$$R(\mathcal{G}) = \frac{1}{T_c} I_s(x_n; y_n)$$

Per esempio, nel caso in cui abbiamo  $y_n = x_n$ , la matrice di probabilità di transizione è uguale alla matrice identità; il tasso di informazione del canale vale  $R(\mathcal{G}) = \frac{1}{T_c} H(x_n)$ .

In generale, per un canale qualsiasi, poichè  $I_s(\underline{x}; \underline{y}) \leq H_s(\underline{x})$ , avremo che

$$R(\mathcal{G}) = \frac{I_s(\underline{x}; \underline{y})}{T_c} \leq \frac{H_s(\underline{x})}{T_c}$$

### 5.1.3 Capacità di un canale numerico M-ario senza memoria

La capacità del canale è il **massimo tasso di informazione del canale** tra tutti quelli ottenuti, usando tutte le possibili distribuzioni della sorgente

$$C = \max(R(\mathcal{G}))$$

Nell'esempio di prima ( $y_n = x_n$ ), la capacità è

$$C = \frac{\log_2(M)}{T_c}$$

(ottenuta con  $x_n$  a valori equiprobabili nell'alfabeto M-ario)

## 5.2 Teorema di Shannon per la codifica di canale

Consideriamo un canale M-ario numerico senza memoria con periodo di simbolo  $T_c$  e capacità  $C$ .

Sia  $\{b_c\}$  una sorgente con tasso di informazione del messaggio nominale  $R$ .

Se  $R < C$ , allora per ogni  $\delta > 0$  e per ogni  $n$  sufficientemente grande, esistono:

1. Un **codice di canale** con parole (con alfabeto M-ario) lunghe  $n$  e con un numero di parole  $\lceil 2^{nRT_c} \rceil$
2. Un **decodificatore per il codice** tali che la probabilità di errore sulla parola di codice è minore di  $\delta$

Il numero di parole di codice dal teorema è

$$2^{nRT_c} = 2^k$$

per un codice a blocco (n,k); si ha quindi che

$$\frac{k}{n} = RT_c < CT_c$$

### 5.2.1 Converse

Per un canale M-ario numerico senza memoria e con capacità  $C$ , esiste  $\delta > 0$  tale che per un qualsiasi codice e strategia di decodifica, se  $R \geq C$  (con  $R$  tasso di informazione nominale della sorgente), la probabilità di errore sulla parola di codice è sempre  $\geq \delta$ .

## 5.3 Canale analogico AWGN

### 5.3.1 Capacità di un canale analogico AWGN

$$r(t) = s_{T_x}(t) + w(t)$$



Con  $s_{Tx}(t)$  segnale analogico e  $w(t)$  rumore AWGN (V.A. Normale con media nulla).

La capacità di questo tipo di canale è

$$C = \frac{1}{T} \log_2 \left( 1 + \frac{\sigma_{s_{Tx}}^2}{\sigma_w^2} \right)$$

con  $\sigma_{s_{Tx}}^2 = E[s_{Tx}^2(t)]$  e  $\sigma_w^2$  la varianza del rumore.

La densità di probabilità dell'ingresso che massimizza la mutua informazione (che raggiunge la capacità calcolata) è Gaussiana a media nulla (per una potenza fissata).

## 5.4 Canale binario simmetrico senza memoria

$$P(\hat{b}_l = 1|b_l = 0) = P(\hat{b}_l = 0|b_l = 1) = P_{bit}$$

(simmetrico)

$$C = \max \left( \frac{I(b_l; \hat{b}_l)}{T_b} \right)$$

L'informazione mutua abbiamo che vale

$$I(b_l; \hat{b}_l) = H(\hat{b}_l) - H(\hat{b}_l|b_l)$$

facendo i calcoli otteniamo che

$$C = \frac{1}{T_C} \{ 1 - [P_{bit} \log_2 \left( \frac{1}{P_{bit}} \right) + (1 - P_{bit}) \log_2 \left( \frac{1}{1 - P_{bit}} \right)] \}$$

## 5.5 Canale binario senza memoria con cancellazione

Si tratta di un canale che ha **in ingresso due possibili simboli** (0, 1) ma **con uscita tre possibili valori** (0, 1, e).

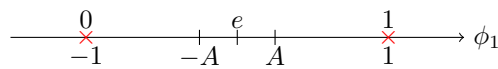
Posso avere una possibilità che avendo trasmesso 1 io riceva e (lo stesso vale per lo 0).

$$P(y = e|x = 0) = P(y = e|x = 1) = \alpha$$

$$P(y = 0|x = 0) = P(y = 1|x = 1) = 1 - \alpha$$

$$P(y = 1|x = 0) = P(y = 0|x = 1) = 0$$

Posso trovare un canale di questo tipo ad esempio se considero una modulazione di canale binaria e sono troppo incerto al ricevitore (se non so cosa ho trasmesso, non mi prendo rischi e non decodifico il messaggio).



$$\alpha = P(w \in [-(1+A); -(1-A)])$$

Essendo però in un canale con rumore gaussiano, non è mai nulla la probabilità che venga ricevuto 0 dato che è stato trasmesso 1, quindi dovremmo (a scopo matematico) introdurre un'altra probabilità:

$$\beta = P(y = 1|x = 0) = P(w > (1+A)) \simeq 0$$

Fissando  $q = P(x = 0)$  abbiamo che:

$$P_y(0) = q(1 - \alpha)$$

$$P_y(1) = (1 - q)(1 - \alpha)$$

$$P_y(e) = \alpha$$

Calcoliamo quindi ora la capacità del canale:

$$C = \frac{1}{T} \max(I(x; y)) = \frac{1 - \alpha}{T}$$

## 6 Codifica di Sorgente

L'operazione di codifica di sorgente, codifica il flusso di informazione e lo converte in un'altra sequenza di simboli.

Lo scopo del processo è **comprimere il messaggio**, in modo da trasmettere il minor numero di *bit/s*.

Dopo la codifica di sorgente, avvengono la codifica di canale, la modulazione, la demodulazione, la decodifica di canale e in fine la decodifica di sorgente.

Ci sono **due tipi di codifiche di sorgente**:

1. **Codifica di sorgente senza perdite (LOSSLESS)**: Il messaggio all'uscita del decodificatore coincide con quello all'ingresso del codificatore.
2. **Codifica di sorgente con perdite (LOSSY)**: Il messaggio all'uscita del decodificatore è simile ma non coincide con quello all'ingresso del codificatore.

### 6.1 Codifica di sorgente senza perdite

La sorgente trasmette da un dizionario  $\mathcal{D}_x$  delle parole  $\underline{x}$  con una densità di probabilità  $p_x$ .

In generale, all'uscita dal codificatore di sorgente, le parole saranno più corte; le parole in uscita allora faranno parte di un alfabeto  $\mathcal{D}_y$  delle parole  $\underline{y}$  con lettere prese da un alfabeto con cardinalità  $M_y$ .

Le parole  $\underline{y}$  hanno una **lunghezza variabile** (in generale), in quanto ho una miglior dinamicità per la creazione delle parole in uscita e non ho le mani legate per come creare il codificatore (possono esserci **parole più corte che corrispondono a parole lunghe più probabili**).

Quello che conta è la lunghezza media delle parole  $\underline{y}$ :

$$L_y = E[L(\underline{y})] = \sum L(\underline{y})p_x(\underline{x}(\underline{y}))$$

con  $\underline{x}(\underline{y})$  si intende la parola  $\underline{x}$  associata alla parola codificata  $\underline{y}$  e con  $L(\underline{y})$  la funzione che mi determina la lunghezza della parola  $\underline{y}$ .

Il codificatore di canale, tuttavia, vuole blocchi di lunghezza fissa, mentre il codificatore di sorgente, cercando di comprimere, trasmette **blocchi con lunghezza variabile**.

Per risolvere a questo problema, prendo le parole trasmesse dal codificatore di sorgente e le unisco in un'unica grande parola; il codificatore di canale poi suddividerà l'unione delle parole in parole della lunghezza necessaria. Se poi non ci saranno errori nel canale sarà possibile ricostruire la sequenza di parole iniziale.

#### 6.1.1 Codici a prefisso

Per riuscire a suddividere le parole che erano state unificate dal codificatore di canale, utilizzo i **codici a prefisso**: un codice di sorgente a prefisso **non ha**

**nessuna parola di codice che è prefisso di un'altra.**

Ad esempio il dizionario  $\mathcal{D}_y = \{0, 10, 111\}$  è un codice a prefisso in quanto ogni parola di codice non è utilizzabile come prefisso di un'altra parola.

## 6.2 Teorema di Kraft Mc-Millan

1. Per ogni codificatore di sorgente senza perdite, deve valere

$$\sum_{b \in \mathcal{D}_y} \frac{1}{M_y^{L_y(\underline{b})}} \leq 1$$

dove  $M_y$  è la cardinalità dell'alfabeto di  $\underline{y}$  e  $L_y(\underline{b})$  è la lunghezza (in simboli) della parola  $\underline{b}$

2. Per una sorgente con dizionario di cardinalità  $N$  (con  $N$  parole da codificare), se  $l_1, \dots, l_N$  sono interi che verificano

$$\sum_{i=1}^N \frac{1}{M_y^{l_i}} \leq 1$$

con  $M_y$  intero, allora esiste un codice a prefisso con alfabeto di cardinalità  $M_y$  e parole di codice di lunghezza  $L_i$

## 6.3 Teorema di Shannon per la codifica di sorgente

1. Per ogni codice di sorgente con alfabeto di cardinalità  $M_y$ , la lunghezza media delle parole soddisfa

$$L_y \geq \frac{H(\underline{X})}{\log_2(M_y)}$$

2. Esiste un codice a prefisso con alfabeto di cardinalità  $M_y$  e lunghezza media delle parole di codice

$$L_y < \frac{H(\underline{x})}{\log_2(M_y)} + 1$$

Il teorema di Shannon **lega la lunghezza media delle parole di codice all'entropia della sorgente.**

Unendo le cose, otteniamo che

$$\frac{H(\underline{x})}{\log_2(M_y)} \leq L_y < \frac{H(\underline{x})}{\log_2(M_y)} + 1$$

sempre se faccio le cose per bene (si tratta di una garanzia di esistenza, non garantisce che ogni codifica di sorgente avrà questo risultato).

Abbiamo quindi che la lunghezza delle parole di codice hanno lunghezza

$$l_i = \lceil \log_{\frac{1}{M_y}}(p_x(a_i)) \rceil \quad \forall a_i \in \mathcal{D}_x$$

Il codice con queste lunghezze si chiama **Codice di Shannon-Fano**.

Il codice di Shannon-Fano è ottimo (raggiunge la lunghezza media minima tra tutti i codici per una data sorgente) se  $p_x(a_i)$  sono **potenze intere** di  $\frac{1}{M_y}$  in modo che il logaritmo dia un numero intero.

$$L_y = E[\log_{\frac{1}{M_y}}(p_x(a_i))] = \frac{H(\underline{x})}{\log_2(M_y)}$$

## 6.4 Codifica di sorgente ottima

Per un codice a prefisso ottimo, **parole a più bassa probabilità hanno lunghezza maggiore**

$$p_x(a_1) \geq p_x(a_2) \rightarrow L(b_1) \leq L(b_2)$$

Nel dizionario di un codice a prefisso ottimo, ci sono almeno due parole di codice  $\underline{b}'$  e  $\underline{b}''$  che hanno lunghezza massima e differiscono solo nell'ultimo simbolo.

## 6.5 Procedura di Huffman

Fornisce un codice di sorgente ottimo data la densità di probabilità del dizionario  $\mathcal{D}_x$  da codificare.

1. Ordiniamo le parole di  $\mathcal{D}_x$  con probabilità decrescente

$$p_x(a'_1) \geq \dots \geq p_x(a'_{N-1}) \geq p_x(a'_N)$$

2. Assegnamo le parole più lunghe a  $a'_{N-1}$  e  $a'_N$  e le costruiamo uguali tranne l'ultimo bit
3. Costruisco un nuovo codice sul dizionario con densità di probabilità  $p(a'_1), \dots, p(a'_{N-1}), p(a'_N)$

## 6.6 Codifica aritmetica

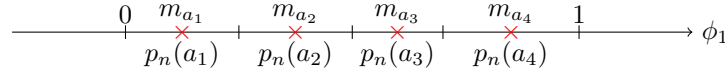
### 6.6.1 Codice di Elias

Serve a fornire un codice a prefisso, non per forza ottimo (ma di più facile implementazione).

Data la densità di probabilità delle parole d'ingresso:

$$\mathcal{D}_n = \{a_1, \dots, a_N\}, \quad p_n(a_1), \dots, p_n(a_N)$$

Disponiamo le probabilità  $p_n$  in sequenza su un segmento lungo 1 e indichiamo con  $m_i$  i punti medi nei segmenti delle probabilità.



Abbiamo quindi che  $m_1 < m_2 < \dots < m_N$ .  
 Convertiamo i numeri in binario  $m_{a_i} = 0.0010..01$  e per ogni parola  $a_i$  calcoliamo la lunghezza della sequenza:  $L_i = \lceil i(a_i) + 1 \rceil$ .  
 Prendiamo quindi la parte dopo la virgola di  $(m_{a_i})_2$  e consideriamo solo i  $L_i$  bit più significativi.  
 Andremo così a creare il codice di Elias che avrà una lunghezza media  $L_y \geq L_y(\text{ottimo})$ .

## 6.7 Efficienza della sorgente

$$\eta = \frac{R(n)}{F \log_2(M)} = \frac{H_s(n)}{\log_2(M)} \quad \eta \in [0, 1], F = \frac{1}{T}$$

### 6.7.1 Efficienza di un codice di sorgente

$$\eta_y = \frac{H(\underline{y})}{L_y \log_2(M_y)} \quad \eta_y \in [0, 1]$$

### 6.7.2 Efficienza del codice ottimo

$$\eta > \frac{H(\underline{n})}{H(\underline{n}) + \log_2(M_y)}$$

1. Per  $M_y = 2$  il limite inferiore è vicino a  $1^-$
2. Vorremmo che  $H(\underline{n}) \rightarrow +\infty$  per ridurre l'effetto di  $\log_2(M_y)$  ma sappiamo essere limitata a  $\log_2(M_n)$ , quindi tenderemo ad **aumentare la dimensione dell'alfabeto**.