# Fondamenti di Telecomunicazioni

Matteo Zanella

A.A 2022-2023

## 1 Segnali

Un segnale è una funzione del tempo $s(t) \in \Re$, 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)

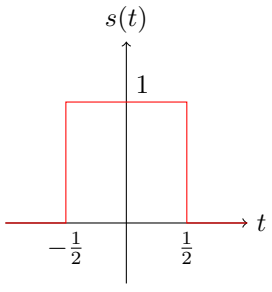
I valori assunti dai segnali possono essere:

1. **continui**: il segnale assume valori $\in \Re$ (o sottoinsiemi)
2. **discreti**: il segnale assume valori discreti

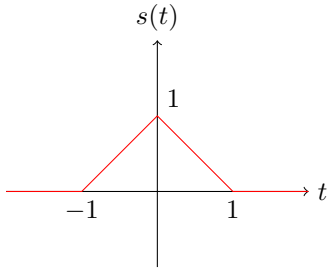
I segnali analogici sono segnali continui con valori continui, mentre i segnali digitali sono segnali discreti con valori discreti.

### 1.1 Esempi di segnali

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



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




## 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:

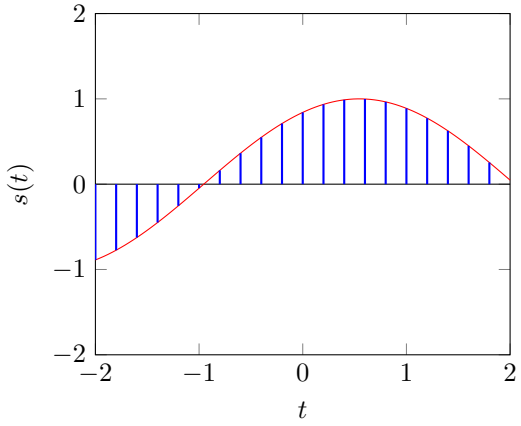
$$s(t) = Ay(t)$$

**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.



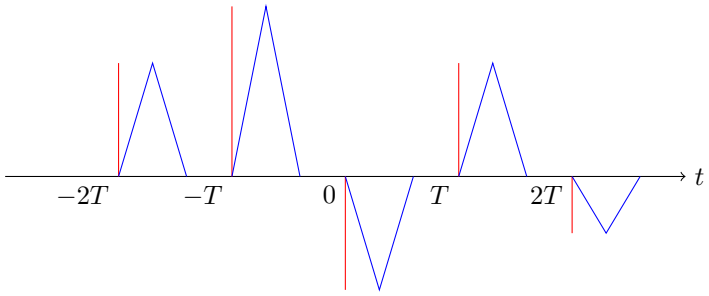

Utilizzare il campionamento provoca una perdita di qualità del segnale.

## 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.

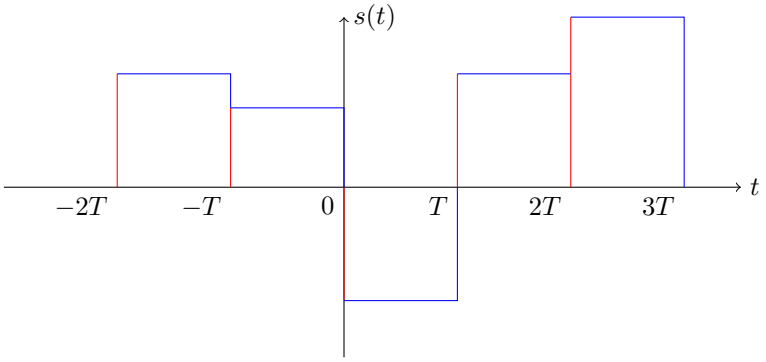



## 1.5 Holder

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:

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

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:

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

# 2 Spazio vettoriale di segnali a energia finita

## 2.1 Prodotto interno

Definiamo il prodotto interno tra due vettori come:

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

## 2.2 Norma di un vettore

Definiamo la norma di un vettore come:

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

## 2.3 Base ortonormale

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

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

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

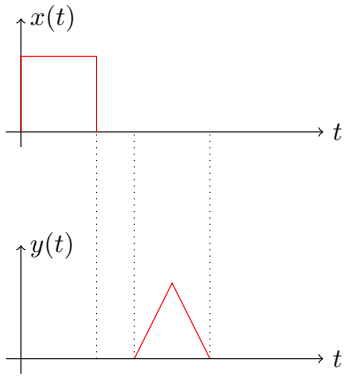
Deve inoltre valere che un segnale generico $y(t)$ sia scrivibile come combinazione lineare dei vettori della base:

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

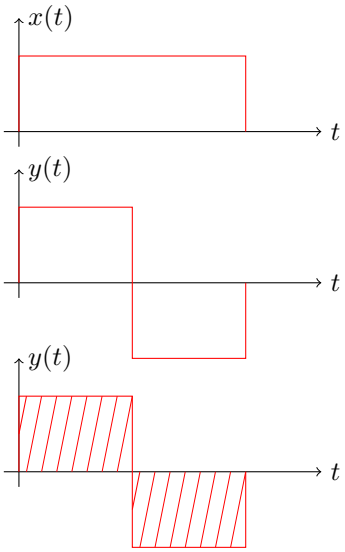
I vettori della base sono **ortogonali** in quanto il loro prodotto interno fa 0 e **normali** in quando il modulo di ognuno di loro vale 1.

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




## 2.4 Metodo di orto normalizzazione di Gram Schmidt

Supponiamo di avere N vettori $s_1(t), ..., s_N(t)$ 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:

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

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

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

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

# 3 Modulazione digitale

## 3.1 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.

## 3.2 Il rumore dopo la proiezione

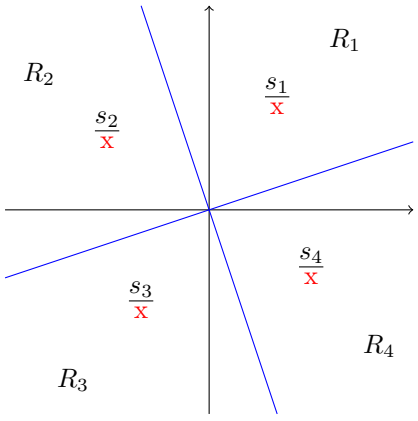
Se trasmettiamo in un canale AWGN, il rumore è rappresentabile come una **v.a. Gaussiana** a media nulla e varianza indipendente dal rumore in altri istanti.

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

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

## 3.3 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**.

## 3.4 Probabilità di decisione corretta

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

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

$N$ è il numero di regioni di decisione, $\Re_n$ 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**: $a_0 = argmax_n \left( p_{a_0|n}(n|r) \right)$ da usare in caso generico
2. **ML**: $a_0 = argmax_n \left( p_{r|a_0}(r|n) \right)$ da usare quando i simboli trasmessi sono equiprobabili
3. **MD**: $a_0 = argmin_n \left( d^2(r, s_n) \right)$ da usare se si hanno simboli equiprobabili trasmessi in un canale AWGN

## 3.5 Ricevitori digitali

1. **MAP-ML**

2. **MD-1**

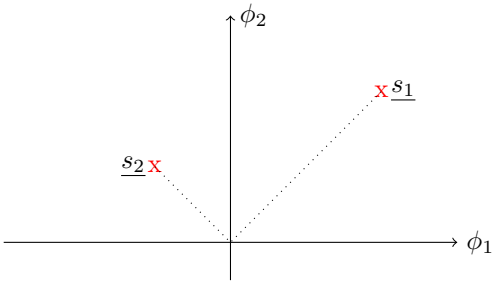
3. **MD-2**

## 3.6 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

$$\underline{s_1} = \left[ \langle\ s_1(t),\ \phi_1(t)\ \rangle, \langle\ s_1(t),\ \phi_2(t)\ \rangle \right]$$

$$\underline{s_2} = \left[ \langle\ s_2(t),\ \phi_1(t)\ \rangle, \langle\ s_2(t),\ \phi_2(t)\ \rangle \right]$$

### 3.6.1 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.

### 3.6.2 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.

### 3.6.3 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:

$$\underline{s_1} = \left[\sqrt{E_{s_1}}, 0\right]$$

$$\underline{s_2} = \left[\rho\sqrt{E_{s_2}}, \sqrt{E_{s_2}}\sqrt{1-\rho^2}\right]$$

## 3.7 Modulazioni M-arie

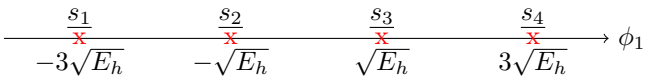
In generale, se abbiamo una segnalazione di **dimensione M**, valgono le seguenti regole:

### 3.7.1 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**:
$$\frac{M-1}{2}Q\left(\sqrt{\frac{E_s}{2\sigma_I^2}}\right)+\frac{1}{M}Q\left(\sqrt{\frac{E_s}{\sigma_I^2}}\right) \leq P(E) \leq (M-2)Q\left(\sqrt{\frac{E_s}{2\sigma_I^2}}\right)+Q\left(\sqrt{\frac{E_s}{\sigma_I^2}}\right)$$

## 3.8 Modulazione PAM

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




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

la base ha dimensione 1 e $\phi(t) = \frac{h_{TX}(t)}{\sqrt{E_h}}$.
La distanza tra due simboli adiacenti è di $2\sqrt{E_h}$.

### 3.8.1 Simboli

$$\underline{s_m} = \alpha_m \sqrt{E_h}$$

### 3.8.2 Energia media

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

### 3.8.3 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**:
$$P(E) = 2Q\left(\sqrt{\frac{E_h}{\sigma_I^2}}\right)$$

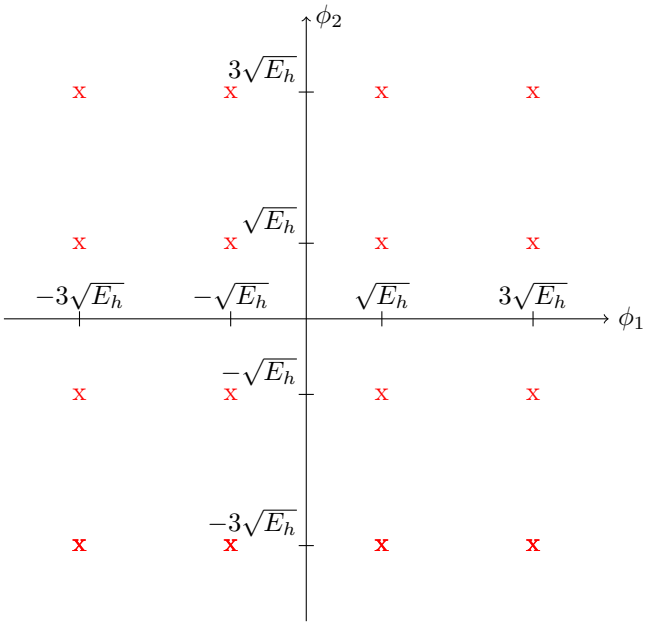
Andando a calcolare la probabilità d'errore totale abbiamo che, se i simboli sono trasmessi con la stessa probabilità:

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

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 $P_{bit} = \frac{P(E)}{\log_2(M)}$

## 3.9 Modulazione QAM

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




esempio di una 16-QAM

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

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

### 3.9.1 Base

Abbiamo come base per la segnalazione:

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

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

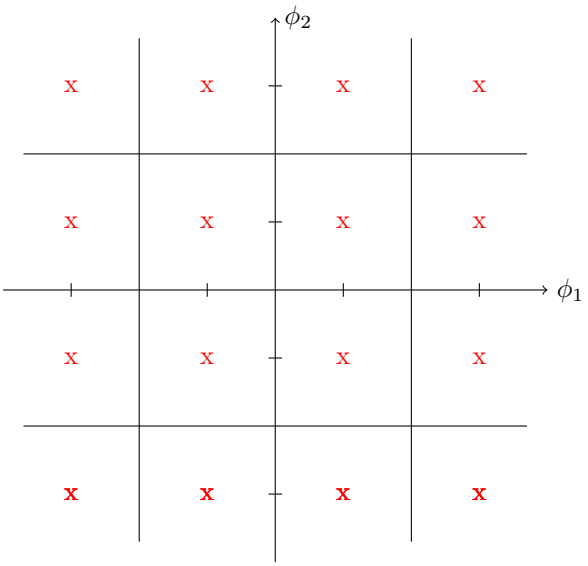
### 3.9.2 Simboli

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

### 3.9.3 Energia media

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

### 3.9.4 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:

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

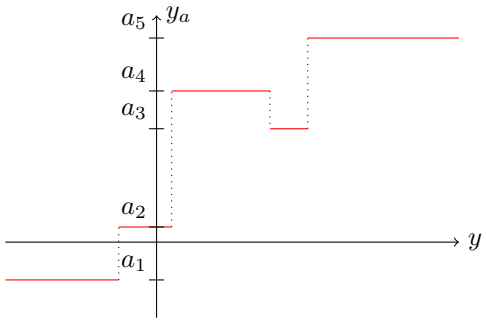
Utilizzando poi la mappatura di Gray per mappare i bit all'interno della segnalazione (due simboli diversi possono differire solo di un bit), otteniamo che la probabilità d'errore sul bit è di $P_{bit} = \frac{P(E)}{\log_2(M)}$

## 3.10 Quantizzazione

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

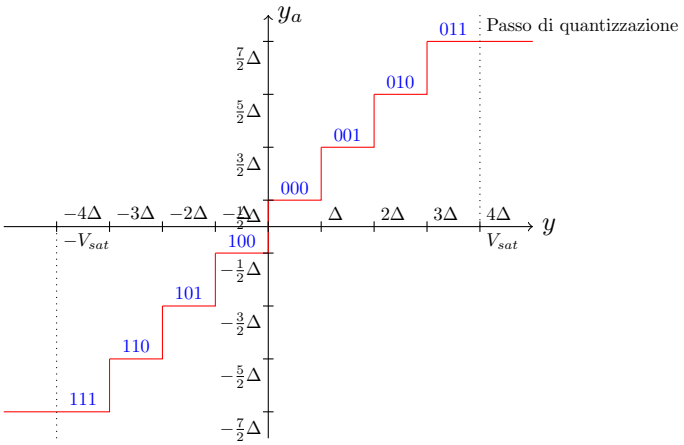
### 3.10.1 Funzione caratteristica

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




### 3.10.2 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 $y \notin [-V_{sat}, V_{sat}]$

### 3.10.3 Gestione dell'errore di saturazione

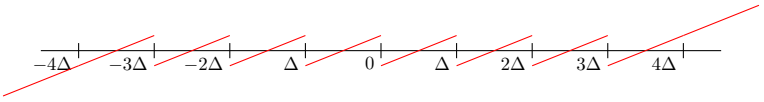
Abbiamo due casi da distinguere:

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:

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

### 3.10.4 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.

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

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 $\sigma_y = \frac{\Delta^2}{12}$

### 3.10.5 Rapporto segnale-rumore

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

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

# 4 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

## 4.1 Protocolli di comunicazione

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

### 4.1.1 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

## 4.2 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.

## 4.3 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**.

### 4.3.1 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 (lunghi $k$) contengono il messaggio che vogliamo trasmettere, quelli di uscita includono anche un'informazione per correggere gli errori.

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

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 $\widetilde{\underline{c}}$ da decodificare, i quali diventeranno di nuovo $2^k$ messaggi decodificati $\hat{\underline{b}}$.

L'insieme dei possibili vettori $\underline{c}$ lo chiamiamo **codice** $\mathcal{C}$, mentre i vettori $\underline{c}$ li chiameremo **parole di codice**.
I vettori $\widetilde{\underline{c}}$ non faranno sempre parte del codice, quindi non sono, in generale, parole del codice ma semplici sequenze binarie.
I bit della sequenza binaria $\widetilde{\underline{c}}$ possono essere scritti come $\widetilde{c_i} = c_i + e_i$, 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 $P_{bit} = P(e_i = 1) \quad \forall i$.

## 4.4 Criteri di decodifica

Come passare da $\hat{\underline{c}}$ a $\hat{\underline{b}}$:
Volendo minimizzare la probabilità di errore sulle sequenza, $P_e = (P(\hat{\underline{b}} \neq \underline{b})$

1. **Criterio MAP**

$$\hat{\underline{c}} = argmax_{\underline{\alpha} \in \mathcal{C}} \big(P(\hat{\underline{c}} = \beta | \underline{c} = \underline{\alpha}) P(\underline{c} = \underline{\alpha})\big)$$

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

$$\hat{\underline{c}} = argmax_{\underline{\alpha} \in \mathcal{C}} \big(P(\underline{c} = \alpha | \hat{\underline{c}} = \underline{\beta}\big)$$

   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

$$\hat{\underline{c}} = argmin_{\underline{\alpha} \in \mathcal{C}} \big(d_H(\underline{\alpha}, \hat{\underline{c}})\big)$$

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

Il numero massimo di errori che posso correggere è $\lfloor \frac{d_H - 1}{2} \rfloor$.
Il numero massimo di errori che posso rilevare è invece di $d_H - 1$.
Con $d_H$ la distanza di hamming tra due parole di codice.

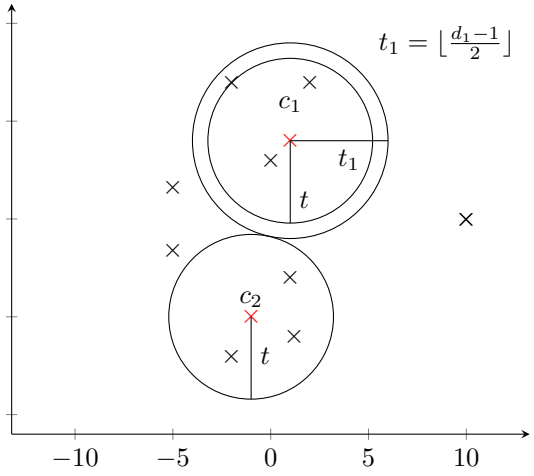
### 4.5.1 Distanza di Hamming minima di un codice $\mathcal{C}$

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

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.

### 4.5.2 Bound di Hamming

Le sequenze che si trovano all'interno di una distanza di $\lfloor \frac{d_{min}-1}{2} \rfloor$ sono sequenza 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 $\frac{k}{n}$

## 4.6 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 $\{0, 1\}$:

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

### 4.6.1 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 $\{\underline{c}_1, ..., \underline{c}_{2^k}\}$ è un sottospazio lineare dello spazio binario di dimensione $n$: $\mathcal{C}$ è lineare $\iff \alpha\underline{c}_1 + \beta\underline{c}_2 \in \mathcal{C} \quad \forall \underline{c}_1, \underline{c}_2$

### 4.6.2 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

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

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

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

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: $d_H(\underline{\gamma}_1, \underline{\gamma}_2) = ||\underline{c}||_H, \quad c \in \mathcal{C}$, quindi abbiamo che

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

### 4.6.3 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

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

$$\underline{c} = \begin{pmatrix} c_1 \\ . \\ c_n \end{pmatrix} \qquad \underline{b} = \begin{pmatrix} b_1 \\ . \\ b_k \end{pmatrix}$$

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

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:

$$\underline{G} = \begin{pmatrix} \underline{I} \\ \hline \underline{A} \end{pmatrix} \qquad \underline{I}[k \times k],\ \underline{A}[(n-k) \times k]$$

generando le parole di codice

$$\underline{c} = \begin{pmatrix} \underline{b} \\ \hline \underline{A}\,\underline{b} \end{pmatrix}$$

### 4.6.4 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.

$$\underline{G} = \begin{pmatrix} \underline{I} \\ \hline \underline{A} \end{pmatrix} \qquad \underline{I}[k \times k],\ \underline{A}[(n-k) \times k]$$

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.

## 4.7 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

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

TODO lezione 26

# 5 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 $P$

## 5.1 Assiomi dell'informazione

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

1. $i(A) \geq 0, \qquad \forall A$
   L'informazione deve esserci per esistere.
2. $P(A) \leq P(B), \qquad i(A) \geq i(B)$
   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:

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

Usando la base 2, l'informazione sarà **rappresentabile in bit**.
Usando la base $e$, l'unità di misura diventa il **Neper**.

## 5.2 Funzione informazione di una variabile aleatoria discreta

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

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

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

Definiamo la **funzione informazione** come

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

### 5.2.1 Informazione di eventi di X

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

Dato un insieme $B \subseteq A_x$,

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

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.

## 5.3 Entropia di una variabile aleatoria

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

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

### 5.3.1 Proprietà dell'entropia

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

## 5.4 Funzioni concave e diseguaglianza di Tensen

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

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

### 5.4.1 Disequazione di Tensen

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

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

## 5.5 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 Tensen:

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

La disequazione vale solo se i valori di $p_x(x)$ sono distinti tra loro.
Se $X$ è uniforme nell'alfabeto ($p_x(a) = \frac{1}{M}$).

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

# 6 Vettori aleatori

Un vettore aleatorio di lunghezza N $\underline{x} = [x_1, ..., x_N],\ n \in [1, N]$ con $x_n$ variabile aleatoria discreta con alfabeto $A_n$ $\underline{x_n} \in A_1 \times ... \times A_N = A$
La densità di probabilità del vettore aleatorio $p_{\underline{x}}(\underline{a}) = P(\underline{x} = \underline{a}), \quad \underline{a} \in A$

## 6.1 Funzione informazione per un vettore aleatorio

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

## 6.2 Entropia di più variabili aleatorie

Consideriamo due v.a. $X$ $Y$, se $Y$ è funzione deterministica di $X$, allora

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

Chiaramente se $X$ è una v.a., anche $f(X)$ lo è

TODO RECUPERO APPUNTI DELLA LEZIONE 06/12
TODO LEZIONE 07/12

## 6.3 Canale numerico M-ario senza memoria

Indichiamo con $T_c$ il tempo di simbolo del canale.
Essendo un canale numerico, $x_n, y_n \in \{a_1, ..., a_M\}$,ed essendo senza memoria, $y_n$ dipende solo da $x_n$ e non dai simboli trasmessi precedentemente.

### 6.3.1 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.

### 6.3.2 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)

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

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

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

### 6.3.3 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

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

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)

## 6.4 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 $\{b_{\mathcal{C}}\}$ 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

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

### 6.4.1 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 $\geq \delta$.