Capitolo 9

Teoria dell'informazione e codifica di sorgente

Come evidenziato fin dal cap. 1, il motivo per voler trasmettere un segnale deriva *dall'informazione* che lo stesso convoglia. Descrivere e misurare questo concetto è indissolubilmente legato alla caratterizzazione della entità che produce il segnale, indicata come *sorgente*, che può essere continua o discreta, con o senza memoria, e che si assume di tipo stazionario, ovvero invariante nel tempo. Dopo aver individuato come misurare l'informazione media (chiamata *entropia*) contenuta nei segnali prodotti da una sorgente, ci occupiamo di rappresentare gli stessi in una forma (chiamata *codifica di sorgente*) in grado di *ridurre* la quantità di dati da trasmettere (ovvero la banda da occupare) senza pregiudicare la qualità del messaggio, ossia senza *perdere* informazione. Mentre nel caso di sorgente discreta è possibile pervenire a soluzioni in tal senso ottimali, per le sorgenti di segnali tempo-continui l'analisi perviene alla possibilità di ridurre *gradualmente* la banda necessaria alla trasmissione, accettando l'insorgenza di una distorsione che rappresenta l'associata perdita di informazione. Approfondimenti sulla codifica di sorgenti multimediali sono svolti al capitolo 10, mentre il cap. 17 determina il massimo tasso informativo che un canale può trasportare, ossia la sua *capacità*, ed illustra come *proteggere* dagli errori di ricezione l'informazione trasmessa, ovvero il tema della *codifica di canale*.

Tipi di sorgente ed elementi che ne consentono la codifica Una sorgente informativa può essere di natura discreta, come nel caso di un documento scritto, o continua, come nel caso di un segnale analogico, ad esempio audio e video. In entrambi i casi, considerazioni di tipo statistico conducono a *misurare* (in bit/simbolo) la quantità media di informazione presente nei messaggi prodotti mediante la definizione di una grandezza, l'*entropia*. Ma allo stesso tempo la descrizione in modo *nativo* di tali messaggi può produrre una *velocità di trasmissione* ben superiore!

La *codifica di sorgente* ha lo scopo di individuare rappresentazioni alternative per le informazioni prodotte dalla sorgente in modo da ridurre il numero di bit/secondo necessari alla trasmissione, e renderlo quanto più possibile prossimo a quello indicato dell'entropia. Ciò avviene sfruttando le caratteristiche della sorgente, del processo di

codifica, e del destinatario dei messaggi, in quanto

- la particolare distribuzione statistica dei simboli o dei valori emessi dalla sorgente può permettere l'uso di un minor numero di bit per rappresentare i simboli *più frequenti* di altri;
- la dipendenza statistica presente tra simboli consecutivi, ovvero la presenza di un fenomeno di *memoria* intrinseco della sorgente, rende possibile *la predizione* (approssimata) dei valori futuri;
- l'introduzione di un *ritardo di codifica* permette di analizzare un intero intervallo temporale del messaggio;
- nel caso di segnali multimediali i *fenomeni percettivi* legati alla fisiologia dell'apparato sensoriale possono guidare il codificatore nella scelta delle componenti di segnale da sopprimere, in quanto percettivamente non rilevanti.

Nel caso di sorgenti nativamente discrete, come ad esempio documenti in formato elettronico, lo scopo della codifica di sorgente è quello di permettere la ricostruzione *integrale* di quanto trasmesso, realizzando una codifica *senza perdita di informazione*. Nel caso invece di sorgenti continue, dove la sequenza numerica è il risultato di un processo di campionamento e quantizzazione, si determina l'insorgenza di una prima causa di *distorsione* nel messaggio ricostruito; la velocità binaria effettiva può quindi essere ulteriormente ridotta grazie allo sfruttamento dei fenomeni percettivi, ed in tal caso il risultato della codifica viene detto *con perdita di informazione*.

9.1 Codifica di sorgente discreta

Iniziamo l'analisi considerando una sorgente di informazione che produce sequenze $x\,(n)$ composte da simboli x_k appartenenti ad un alfabeto di cardinalità L (ossia con $k = \{1, 2, \cdots, L\}$), e che si presentano con probabilità $p_k = Pr\,(x_k)$ non dipendente da n, ovvero la sorgente è stazionaria.

Sorgente senza memoria Con questo termine si intende che i simboli vengono emessi in modo *statisticamente indipendente* (§ 6.1.5), ovvero indicando con x_h, x_k una coppia di simboli consecutivi (ossia $x\,(n) = x_h$, $x\,(n+1) = x_k$), la probabilità del secondo non dipende dall'identità del primo, ossia $p\,(x_k/x_h) = p\,(x_k) = p_k$.

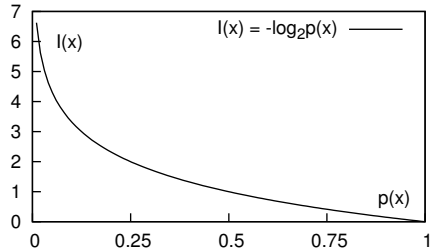
Misura dell'informazione La conoscenza di ognuno dei simboli emessi x_k apporta una quantità di informazione (espressa in *bit*) definita come[1]

$$I_k = I\,(x_k) = \log_2 \frac{1}{p_k} = -\log_2 p_k \text{ bit} \tag{9.1}$$

che rappresenta il *livello di dubbio* a riguardo del verificarsi dell'evento x_k prima che

[1] Per calcolare il logaritmo in base 2, sussiste la relazione $\log_2 \alpha = \frac{\log_{10} \alpha}{\log_{10} 2} \simeq 3.322 \cdot \log_{10} \alpha$. O più in generale, $\log_2 \alpha = \frac{\log_\beta \alpha}{\log_\beta 2}$

questo si verifichi, ovvero di quanto possiamo ritenerci sorpresi nel venire a conoscenza dell'evento x_k, di cui riteniamo di conoscere la probabilità p_k. Osserviamo infatti che la (9.1) attribuisce un valore di informazione tanto più elevato quanto minore è la probabilità di emissione del simbolo.



La scelta di esprimere la relazione tra probabilità e informazione mediante il logaritmo in base 2 consente di verificare le seguenti osservazioni:

Prob. p_k	Inf. $-\log_2 p_k$	Commento
1	0	L'evento certo non fornisce informazione
0	∞	L'evento impossibile dà informazione infinita
$\frac{1}{2}$	1	Conoscere quale tra due eventi equiprobabili si sia verificato apporta un'informazione pari ad una cifra binaria (0/1) o **bit** = *binary digit*
$1/2^n$	n	Es. probabilità $1/4 \rightarrow$ due bit, $1/8 \rightarrow$ tre bit ...

Notiamo inoltre che, essendo la sorgente senza memoria, due simboli emessi consecutivamente sono statisticamente indipendenti ovvero $p(x_h x_k) = p(x_h)\, p(x_k)$ e dunque $I(x_h, x_k) = -\log_2 p_h p_k = -\log_2 p_h - \log_2 p_k = I(x_h) + I(x_k)$.

9.1.1 Entropia

Come in termodinamica al concetto di entropia si associa il grado di *disordine* in un sistema, così per una sorgente informativa l'entropia misura il livello *medio* di *casualità* dei simboli emessi. Definiamo infatti *entropia* (indicata con H) di una sorgente discreta S *il valore atteso* (§ 6.2.2) della quantità di informazione apportata dalla conoscenza dei simboli (scelti tra L possibili) da essa generati

$$H_s = E\{I_k\} = \sum_{k=1}^{L} p_k I_k = \sum_{k=1}^{L} p_k \log_2 \frac{1}{p_k} \text{ bit/simbolo} \tag{9.2}$$

che, pesando in probabilità la quantià di informazione associata ai diversi simboli, rappresenta il *tasso medio* di informazione per simbolo espresso dalle sequenze osservabili. Come dimostriamo sotto, da tale definizione ne consegue che

- se i simboli sono *equiprobabili* ($p_k = \frac{1}{L}$ con $\forall k$) la sorgente è *massimamente informativa*, e la sua entropia è la massima possibile per un alfabeto ad L simboli, pari a $H_{s_{Max}} = \frac{1}{L} \sum_{k=1}^{L} \log_2 L = \log_2 L$ bit/simbolo;
- se i simboli non sono equiprobabili, allora $H_s < \log_2 L$;
- se la sorgente emette sempre e solo lo stesso simbolo, allora $H_s = 0$.

Questi predicati possono essere riassunti dall'espressione

$$0 \leq H_s \leq \log_2 L \tag{9.3}$$

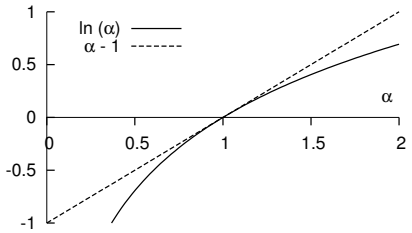
Dimostrazione Osserviamo innanzitutto che $H_s \geq 0$ in quanto la (9.2) comprende tutti termini positivi o nulli, essendo $\log_2 \alpha \geq 0$ per $\alpha = 1/p_k \geq 1$. Mostriamo ora che

$H_s - \log_2 L \leq 0$: riscriviamo innanzitutto il primo membro della diseguaglianza come

$$H_s - \log_2 L = \sum_k p_k \log_2 \frac{1}{p_k} - \log_2 L \cdot \sum_k p_k = \sum_k p_k \left(\log_2 \frac{1}{p_k} - \log_2 L \right) = \sum_k p_k \log_2 \frac{1}{L \cdot p_k} \tag{9.4}$$

dato che $\sum_k p_k = 1$, ove le sommatorie su k si intendono da 1 ad L. Esprimiamo poi questo risultato parziale nei termini di logaritmi *naturali*, tenendo conto che $\log_2 \alpha = \frac{\ln \alpha}{\ln 2}$, ovvero

$$\sum_k p_k \log_2 \frac{1}{L \cdot p_k} = \frac{1}{\ln 2} \sum_k p_k \ln \frac{1}{L \cdot p_k} \tag{9.5}$$



A questo punto utilizziamo la relazione

$$\ln \alpha \leq \alpha - 1$$

mostrata in figura, con l'uguaglianza valida solo se $\alpha = 1$.

Ponendo quindi $\alpha = \frac{1}{L \cdot p_k}$ e sostituendo la (9.5) nella (9.4) si ottiene

$$\begin{aligned} H_s - \log_2 L &= \frac{1}{\ln 2} \sum_k p_k \ln \frac{1}{L \cdot p_k} \leq \frac{1}{\ln 2} \sum_k p_k \left(\frac{1}{L \cdot p_k} - 1 \right) = \\ &= \frac{1}{\ln 2} \left(\sum_k \frac{1}{L} - \sum_k p_k \right) = \frac{1}{\ln 2} (1 - 1) = 0 \end{aligned}$$

con il segno di uguale solo se $\frac{1}{L \cdot p_k} = 1$ ovvero $p_k = \frac{1}{L}$.

9.1.1.1 Entropia di sorgente binaria

Nel caso particolare di una sorgente *binaria*, ovvero che emette uno tra due simboli $\{x_0, x_1\}$ con probabilità rispettivamente $p_0 = p$, $p_1 = q = 1 - p$, la formula dell'entropia (9.2) fornisce l'espressione

$$H_b(p) = -p \log_2 p - (1 - p) \log_2 (1 - p) \text{ bit/simbolo} \tag{9.6}$$

il cui grafico è mostrato in figura 9.1, in funzione di p.

I due simboli $\{x_0, x_1\}$ possono essere rappresentati dalle 2 cifre binarie $\{0, 1\}$, che in questo caso chiamiamo *binit*, per non confonderli con la misura dell'informazione (il bit). Osserviamo quindi che se $p \neq 0.5$ si ottiene $H_b(p) < 1$, ossia la sorgente emette informazione con un tasso inferiore a un bit/simbolo, mentre a prima vista non potremmo usare meno di un binit per rappresentare ogni simbolo binario.

9.1.1.2 Ridondanza

Esprime la differenza D tra l'entropia di una sorgente H_s (9.2) ad L simboli ed il numero di binit[2] $M = \lceil \log_2 L \rceil$ necessario a rappresentarli, divisa per quest'ultimo, ovvero

$$D = \frac{M - H_s}{M} = 1 - \frac{H_s}{M} \leq 1 \tag{9.7}$$

[2]La notazione $\lceil \alpha \rceil$ indica l'intero superiore ad α: ad esempio con $L = 10$ occorrono $M = \lceil \log_2 10 \rceil = \lceil 3.322 \rceil = 4$ binit/simbolo, come se fosse stato $L = 16$.

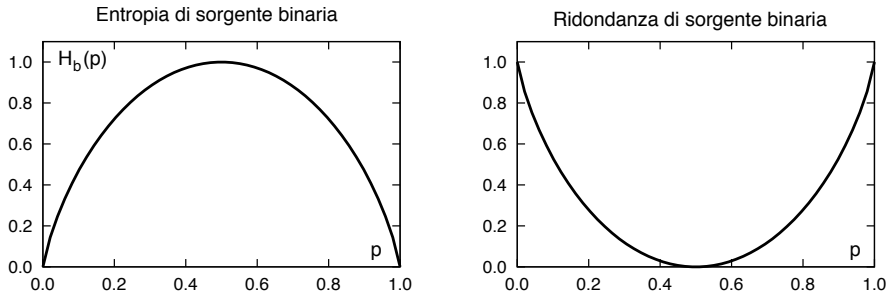


Figura 9.1: Entropia di sorgente binaria, e ridondanza associata

mostrata sempre in fig. 9.1 per il caso $L = 2$ al variare di p[3].

Esempio Consideriamo un sorgente binaria con $p_0 = 0.8$ (e $p_1 = 0.2$). L'applicazione della (9.6) fornisce un valore $H_b\,(0.8) = 0.8 \log_2 \frac{1}{0.8} + 0.2 \log_2 \frac{1}{0.2} = 0.72$ bit/simbolo, minore del valore di 1 bit/simbolo che si sarebbe ottenuto nel caso di equiprobabilità. La relativa ridondanza D è pari a $1 - {}^{0.72}/_{1} = 0.28$, ovvero il 28 %.

9.1.1.3 Entropia di sorgente L-aria

L'applicazione della (9.3) al caso di una sorgente che emette simboli *non* equiprobabili ed appartenenti ad un alfabeto di cardinalità L, determina per la stessa un valore di entropia $H_L < \log_2 L$ bit/simbolo.

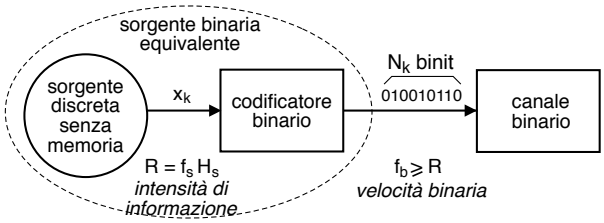
Esempio Nel caso di una sorgente quaternaria con $p_0 = 0.5$, $p_1 = 0.25$, $p_2 = 0.125$, $p_3 = 0.125$, l'applicazione della (9.2) fornisce $H_4 = 1.75$ bit/simbolo, inferiore ai 2 bit/simbolo di una sorgente con quattro simboli equiprobabili. La relativa ridondanza è ora pari a $1 - {}^{1.75}/_{2} = 0.125$ ovvero il 12.5 %.

9.1.2 Intensità informativa e codifica binaria

Svolgiamo ora alcune considerazioni relative alla possibilità di ridurre la ridondanza mediante una operazione di *codifica* (di sorgente). Consideriamo una sorgente discreta senza memoria con alfabeto ad L simboli, caratterizzata da una entropia di H_s bit/simbolo, e che emette i valori x_k a frequenza f_s simboli/secondo: il *flusso informativo* risultante consegue quindi una *intensità* o *velocità di informazione* pari a

$$R = f_s \cdot H_s \quad \text{bit/secondo} \tag{9.8}$$

Volendo trasmettere tale informazione attraverso un *canale binario* (vedi § 17.1.1), l'elemento indicato in figura come *codificatore binario* fa corrispondere ad ogni simbolo x_k un numero va-



[3] Si noti la differenza: la ridondanza della codifica *di sorgente* indica la frazione di binit/simbolo che eccedono il valore dell'entropia, mentre la ridondanza della codifica *di canale* (pag. 469) indica il rapporto tra binit di protezione e quelli di effettivamente emessi dalla sorgente.

riabile di N_k binit[4], scelti in modo da utilizzare meno binit per i simboli più probabili (e più binit per quelli *rari*) come descritto nel seguito, producendo una *velocità di trasmissione* binaria di f_b $^{binit}/_{sec}$. Dal punto di vista del canale il messaggio è prodotto da una nuova sorgente *equivalente*, i cui simboli binari hanno probabilità p e $1-p$, e dunque caratterizzata da una entropia $H_b(p)$ $^{bit}/_{binit} \leq 1$. Dato che l'intensità informativa in ingresso $f_s \cdot H_s$ ed in uscita $f_b \cdot H_b(p)$ dal codificatore deve essere la stessa[5] e che $H_b(p) \leq 1$, la velocità binaria f_b della sorgente binaria equivalente rispecchia il vincolo

$$f_b \geq f_b \cdot H_b(p) = f_s \cdot H_s = R \tag{9.9}$$

Il rapporto $\bar{N} = \frac{f_b}{f_s} \geq H_s$ rappresenta il numero *medio* di binit emessi per ciascun simbolo della sorgente, e può essere valutato a partire dalle probabilità p_k dei simboli x_k e dal numero N_k di binit necessario a rappresentarlo, come valore atteso $\bar{N} = E\{N_k\} = \sum_k p_k N_k$.

9.1.2.1 Teorema della codifica di sorgente

Noto anche come *primo teorema di Shannon*[6], afferma che *esiste* un modo di scegliere gli N_k binit associati a ciascun simbolo x_k tale che[7]

$$H_s \leq \bar{N} \leq H_s + \epsilon \tag{9.10}$$

con ϵ piccolo a piacere, e che si annulla in corrispondenza della codifica *ottima*, per la quale risulta $\bar{N} = H_s$. Ma non dice come fare, cosa di cui ci occupiamo ai §§ seguenti.

9.1.2.2 Codebook e codeword

Le operazioni svolte dal codificatore binario sono descritte nei termini della emissione di una parola di codice detta anche *codeword*, prelevata da un dizionario (o *codebook*) che descrive la collezione di tutte le possibili codeword.

9.1.2.3 Efficienza del codice

E' la misura η di quanti bit di informazione sono trasportati da ogni binit di codifica[8], ed è definita come il rapporto tra l'entropia di sorgente H_s $^{bit}/_{simbolo}$ ed il numero medio $\bar{N}$ di $^{binit}/_{simbolo}$ emessi: in base alla (9.8) ed alla considerazione che $f_b = f_s \cdot \bar{N}$ si ottiene

$$\eta = \frac{H_s}{\bar{N}} = \frac{H_s}{f_b/f_s} = \frac{f_s \cdot H_s}{f_b} = \frac{R}{f_b} = H_b(p) \leq 1 \quad \left[{}^{bit}/_{binit}\right] \tag{9.11}$$

[4] Mettere in corrispondenza i diversi simboli di sorgente con una loro codifica binaria è detta *codifica per blocchi*, discussa al § 9.1.4, dove si mostra anche la possibilità di produrre ogni parola di uscita in corrispondenza non di un *unico* simbolo di sorgente alla volta, ma come equivalente di più simboli. Raggruppando ad esempio M simboli *binari* si ottiene una nuova sorgente equivalente con $L' = 2^M$ simboli.

[5] Essendo biunivoca la corrispondenza tra il simbolo x_k ed il gruppo di N_k binit, non vi è perdita o aggiunta di informazione.

[6] Vedi ad es. `http://it.wikipedia.org/wiki/Primo_teorema_di_Shannon`

[7] In effetti la (9.10) sussiste qualora il codificatore non operi indipendentemente su ogni simbolo di sorgente, ma più in generale possa emettere i binit in corrispondenza di sequenze di x_k *via via più lunghe*. Torneremo su questo aspetto al § 9.1.4, dove il teorema sarà dimostrato.

[8] Ad esempio, un valore $\eta = 0.33$ indica che ogni binit trasporta solo 1/3 di bit di informazione.

e pertanto η è anche pari al rapporto tra gli R $^{bit}/_{secondo}$ di informazione della sorgente e la velocità di trasmissione f_b $^{binit}/_{secondo}$ prodotta dal codificatore.

Osservazione All'aumentare dell'efficienza, si assiste ad una contemporanea riduzione della ridondanza, potendo scrivere[9] $\eta + D = 1$.

Sappiamo già che qualora i binit emessi a velocità f_b assumano i valori 0 o 1 in modo equiprobabile, allora per la sorgente equivalente risulta $H_b\left(\frac{1}{2}\right) = 1$, ovvero dalla (9.9) $f_b = R$ e dalla (9.11) $\bar{N} = H_s$. Dunque il problema di individuare un codice ottimo diviene quello di trovare un insieme di *codeword* tali da rendere *equiprobabili* i valori dei binit, con il vincolo di mantenere il codice *decifrabile*, ovvero tale da rispettare la *regola del prefisso*. Ma andiamo con ordine.

9.1.3 Codifica con lunghezza di parola variabile

Mostriamo mediante un esempio come scegliendo codeword *più lunghe* per rappresentare i simboli *meno probabili*, e *più corte* per i simboli *più frequenti,* si può subito ottenere una migliore efficienza del codice. Consideriamo infatti la sorgente del secondo esempio a pag. 253, con alfabeto di cardinalità $L = 4$, ai cui simboli competono le probabilità riportate alla seconda colonna della tabella. In questo caso l'entropia vale

Simbolo	Prob.	Codeword	N_k
x_1	.5	0	1
x_2	.25	10	2
x_3	.125	110	3
x_4	.125	111	3

$$\begin{aligned} H_s &= \sum_k p_k \log_2 \tfrac{1}{p_k} = \\ &= \tfrac{1}{2}\log_2 2 + \tfrac{1}{4}\log_2 4 + \tfrac{2}{8}\log_2 8 = \tfrac{1}{2} + \tfrac{1}{2} + \tfrac{2}{8}\cdot 3 = 1.75 \ \textit{bit/simbolo} \end{aligned}$$

Se il codificatore di sorgente adotta le codeword mostrate nella terza colonna, a cui corrispondono le lunghezze di N_k binit riportate nella quarta colonna, il numero *medio* di binit/simbolo prodotti dalla codifica binaria risulta pari a

$$\overline{N} = E\{N_k\} = \sum_k N_k p_k = 1\cdot\tfrac{1}{2} + 2\cdot\tfrac{1}{4} + 3\cdot\tfrac{2}{8} = 1.75 \ \textit{binit/simbolo} \tag{9.12}$$

Con queste codeword otteniamo dunque $H_s = \bar{N}$, ovvero una efficienza $\eta = 1$! Intraprendiamo allora un ragionamento che ci porterà a concludere come questo sia un risultato per nulla scontato, e che dipende sia dalla particolare scelta fatta per le codeword, sia dal particolare tipo delle p_k dell'esempio, tutte potenze negative di due (essendo $0.5 = 2^{-1}$, $0.25 = 2^{-2}$, $0.125 = 2^{-3}$).

9.1.3.1 Regola del prefisso

Affinché un insieme di codeword di lunghezza variabile possa assere adottato come *codebook di sorgente*, queste devono poter essere riconosciute come *distinte* presso il ricevitore, e ciò è possibile a patto che nessuna sia *uguale all'inizio* di una codeword più lunga. Si può mostrare che la condizione necessaria e sufficiente per avere un codice *non ambiguo* è che il numero di binit N_k con cui sono espresse le codeword soddisfi la

[9] Sebbene la (9.7) esprima la ridondanza come $D = 1 - \frac{H_s}{M}$, dopo la codifica i simboli di sorgente sono rappresentati (in media) da $\bar{N}$ binit anziché M, dunque otteniamo $\eta + D = \frac{H_s}{\bar{N}} + 1 - \frac{H_s}{\bar{N}} = 1$.

disuguaglianza di Kraft[10], espressa come

$$K = \sum_{k=1}^{L} 2^{-N_k} \leq 1 \tag{9.13}$$

Esempio Nella tabella sono riportati quattro possibili codici (*A,B,C,D*) per la sorgente quaternaria già discussa, assieme al corrispettivo valore di $\bar{N}$ e K.

Simb.	p_k	*A*	*B*	*C*	*D*
x_1	.5	00	0	0	0
x_2	.25	01	1	01	10
x_3	.125	10	10	011	110
x_4	.125	11	11	0111	111
$\bar{N}$		2.0	1.25	1.875	1.75
K		1.0	1.5	0.9375	1.0

Il codice *A* corrisponde ad un codificatore particolarmente banale con $N_k = \bar{N}$ per tutti i k, dunque la (9.13) diviene $K = L2^{-\bar{N}} \leq 1$ ed è soddisfatta a patto che $\bar{N} \geq \log_2 L$: nel nostro caso, essendo $L = 4$ ed $\bar{N} = 2$, si ottiene $\log_2 L = 2 = \bar{N}$ e quindi $K = 1$, dunque il codice è decifrabile (anche perché a lunghezza fissa), ma non particolarmente valido, in quanto l'efficienza espressa dalla (9.11) risulta pari a $\eta = \frac{H_s}{\bar{N}} = \frac{1.75}{2} = 0,875 < 1$. Ma quando $H_s < \log_2 L$ come nel nostro caso, si può realizzare una efficienza migliore ricorrendo ad un codice a lunghezza variabile.

Le codeword del codice *B* producono un valore $K = 1.5 > 1$, e dunque rappresentano un codice ambiguo[11]: difatti, violano la regola del prefisso. Il codice *C* invece non è ambiguo[12], essendo $K < 1$, ma presenta una efficienza $\frac{H_s}{\bar{N}} = 0,9\bar{3} < 1$ e dunque è anch'esso sub-ottimale. Infine, il codice *D* è quello analizzato al precedente paragrafo, ed effettivamente risulta una scelta *ottima*, dato che oltre a soddisfare la (9.13), consegue una efficienza $\frac{H_s}{\bar{N}} = 1$.

9.1.3.2 Codice ottimo

Indichiamo con questo termine un codice che oltre a soddisfare la regola del prefisso[13] consegue anche una efficienza (9.11) unitaria, ovvero $\frac{H_s}{\bar{N}}$=1. Perché ciò avvenga il valore p_k delle probabilità di simbolo *deve* essere una potenza negativa di due, ovvero $p_k = 2^{-N_k}$ con N_k intero: in tal caso la (9.13) si scrive $K = \sum_{k=1}^{L} 2^{-N_k} = \sum_{k=1}^{L} p_k = 1$ e la diseguaglianza di Kraft è verificata con il segno di uguale, dunque è possibile individuare un codice non ambiguo.

Osserviamo inoltre che scegliendo la lunghezza delle codeword proprio pari a $N_k = \log_2 \frac{1}{p_k}$, l'espressione (9.12) che ne calcola la lunghezza media $\bar{N} = \sum_k p_k N_k$ coincide con quella (9.2) che fornisce $H_s = \sum_k p_k \log_2 \frac{1}{p_k}$, ovvero ogni simbolo è codificato con una codeword lunga tanti binit quanti sono i bit di informazione che trasporta, determinando una efficienza η unitaria. Per individuare un codice che *si avvicini* a questa proprietà si può utilizzare la tecnica di *Huffman* presentata appresso, mentre per *modificare* le p_k si ricorre alla *codifica per blocchi* di simboli esposta al § 9.1.4.

[10] Vedi ad es. `https://en.wikipedia.org/wiki/Kraft-McMillan_inequality`

[11] Ad esempio, la sequenza 10110010 potrebbe essere interpretata come $x_3x_4x_1x_1x_3$ oppure $x_2x_1x_4x_1x_1x_2x_1$ od anche $x_3x_2x_2x_1x_1x_3$

[12] Nonostante il codice *C* non soddisfi la regola del prefisso, non è ambiguo in quanto lo zero indica comunque l'inizio di una *nuova* codeword.

[13] Soddisfare la (9.13) con il segno di uguale è una condizione solamente necessaria, ma non sufficiente, per ottenere di un codice ottimo.

9.1.3.3 Codice di Huffman

E' basato su di un *algoritmo* capace di individuare un codice a lunghezza variabile che soddisfa la regola del prefisso, adotta codeword più lunghe per i simboli meno probabili, e tenta di rendere equiprobabili le cifre binarie che compongono le codeword. L'algoritmo definisce[14] un *albero binario* i cui rami sono etichettati con 1 e 0, che può essere realizzato attuando i seguenti passi:

- crea una lista contenente i simboli della sorgente, ordinati per valore di probabilità decrescente, ed associa ad ognuno di essi un nodo-foglia dell'albero;
- finché c'è più di un nodo nella lista:
  - rimuovi dalla lista i due nodi con la probabilità più bassa;
  - crea un nuovo nodo interno all'albero con questi due nodi come figli, e con probabilità pari alla somma delle loro probabilità;
  - aggiungi il nuovo nodo alla lista, in ordine di probabilità;
- il nodo rimanente è la radice, e l'albero è completo;
- assegna cifre binarie diverse ad ogni coppia di rami a partire dalla radice, concatenando le quali si ottengono le codeword per i simboli sulle foglie

Si può dimostrare che il codice di Huffman generato in questo modo è il migliore possibile nel caso in cui la statistica dei simboli di sorgente sia nota a priori, nel senso che produce un codebook con il minor numero possibile di binit/simbolo medi $\bar{N}$, e le cui codeword allo stesso tempo soddisfano la regola del prefisso e la disuguaglianza di Kraft. La codifica di Huffman è ampiamente utilizzata nel contesto di altri metodi di compressione (metodo DEFLATE di PKZIP, § 9.2.3) e di codec multimediali (JPEG e MP3, cap. 10), in virtù della sua semplicità, velocità, ed assenza di brevetti.

Ovviamente ci deve essere un accordo a priori tra sorgente e destinatario a riguardo della corrispondenza tra parole di codice e simboli (o blocchi di simboli) della sorgente. Nel caso in cui ciò non sia vero, oppure nel caso in cui la statistica dei simboli della sorgente sia *stimata* a partire dal materiale da codificare, occorre inviare all'inizio della comunicazione anche la tabella di corrispondenza, eventualmente in forma a sua volta codificata.

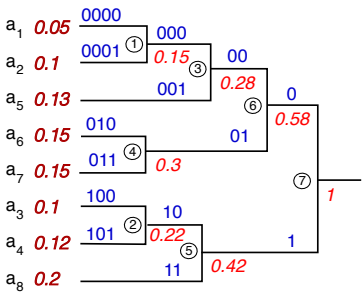
Esempio Una sorgente con $L = 8$ simboli è caratterizzata dalle probabilità di simbolo riportate alla figura seguente, a partire dalle quali si realizza un codice di Huffman mediante la costruzione grafica riportata, in cui le probabilità sono scritte *in rosso*, *sotto* i rami ed accanto ai simboli, mentre i binit *sopra* i rami, *in blu*.

Dopo aver ordinato i simboli in base alle probabilità, si individuano i due nodi con probabilità più bassa come a_1 e a_2, che assommano prob. 0.15 e sono etichettati come nodo ①; dunque la coppia ora meno probabile è a_3 con a_4, che cumula prob. 0.22 e si etichetta ②. Quindi, le due prob. minori divengono quelle del nodo ① e del simbolo a_5, che assommano a 0.28 generando il nodo ③; il passo successivo è quello di accoppiare a_6 con a_7 generando il nodo ④ a cui compete la prob. di 0.3.

Gli ultimi tre passi vedono accoppiare ② con a_8 producendo ⑤ con prob. 0.42,

[14] Vedi `http://en.wikipedia.org/wiki/Huffman_coding`

Simbolo	p_k	Codeword
a_1	0.05	0000
a_2	0.1	0001
a_3	0.1	100
a_4	0.12	101
a_5	0.13	001
a_6	0.15	010
a_7	0.15	011
a_8	0.2	11



quindi ③ con ④ generando ⑥ con probabilità 0.58, ed infine ⑤ con ⑥ producendo il nodo radice a cui compete una prob. unitaria.

Si può ora procedere, partendo dalla radice a destra, ad assegnare un binit pari a 0 o 1 ad ogni coppia di rami rispettivamente in alto ed in basso, ripetendo l'assegnazione seguendo le diramazioni verso sinistra, sopra le quali sono mostrate le codeword che si formano, di cui l'inizio in comune replica la configurazione assegnata al padre. Le codeword complete che compaiono sui rami più a sinistra sono quindi riportate alla terza colonna della tabella. Come è possibile verificare, il codice rispetta la regola del prefisso, in quanto nessuna delle codeword è uguale all'inizio di altre.

Osservazioni Se calcoliamo H_s, $\bar{N}$ e η per la sorgente ed il codice individuato, si ottiene $H_s = 2.916$, non molto meno del massimo $\log_2 L = 3$ bit a simbolo, corrispondente a probabilità p_k tutte uguali e pari a $1/8 = 0.125$. Il codice consegue una *lunghezza media* di codeword pari a $\bar{N} = 2.95$, e come osserviamo usa 3 binit per i 5 simboli con probabilità intermedia, il 15% delle volte usa 4 binit, ed il 20% due. Il codice consegue pertanto una *efficienza* $\eta = H_s/\bar{N} = 0.988$, ovvero la sorgente codificata presenta una *ridondanza* D solamente del 1,2%.

9.1.3.4 Codifica dinamica (di Huffman)

L'esecuzione dell'algoritmo di Huffman richiede la *preventiva* stima della probabilità p_k dei simboli, ottenuta a partire dal messaggio da codificare; inoltre, prima di trasmettere il messaggio codificato occorre inviare anche il codebook prodotto dall'algoritmo, per permettere al decodificatore di funzionare.

La variante *adattiva* dell'algoritmo di generazione del codice prevede invece di utilizzare valori p_k stimati *durante* l'analisi del messaggio, con quelle $\hat{p}_k$ costruire l'albero, e con l'albero *corrente* codificare i simboli man mano che vengono presi in considerazione. Durante l'analisi e la codifica del messaggio le $\hat{p}_k$ si modificano, con esse l'albero, ed il codice. Lo stesso algoritmo adattivo è implementato anche al ricevitore, che sviluppa dal suo lato il medesimo albero, evitando così di dover trasmettere il codebook, e permettendo di adottare la tecnica anche per messaggi prodotti in *tempo reale*. Inoltre, nel caso in cui il messaggio non sia propriamente stazionario e le p_k non si mantengano costanti nel tempo, l'adattività consente al codice di *seguire* tale variazioni e di conseguire in tal caso prestazioni anche migliori della tecnica *statica*[15].

[15] Per approfondimenti si veda

9.1.4 Codifica per blocchi

Riprendiamo la discussione iniziata a pag. 256 relativa al codice binario ottimo per una sorgente $L-$aria senza memoria con simboli x_k a probabilità p_k, notando che se la lunghezza N_k della codeword associata ad x_k viene scelta in modo tale che

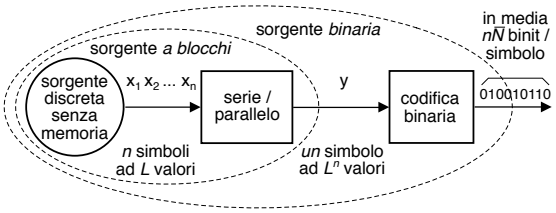
$$\log_2 \frac{1}{p_k} \leq N_k \leq \log_2 \frac{1}{p_k} + 1 \tag{9.14}$$

si può mostrare che la disuguaglianza di Kraft (9.13) è soddisfatta[16], e dunque è possibile realizzare un codice *non ambiguo* con tali codeword. Moltiplicando ora i membri di (9.14) per p_k e sommando su k si ottiene

$$H_s \leq \bar{N} \leq H_s + 1 \tag{9.15}$$

da cui si deduce che è possibile ottenere un'efficienza $\eta = \frac{H_s}{\bar{N}} \to 1$ solo se $H_s \gg 1$, oppure se $N_k \simeq \log_2 \frac{1}{p_k}$ (vedi nota 16).

Ma esiste anche un'altra possibilità: quella di raggruppare i simboli x_k in blocchi di n elementi, e considerare l'intero blocco come un *unico simbolo* di una nuova sorgente equivalente con alfabeto a L^n valori[17], come rappresentato in figura. Essendo la sorgente senza memoria i suoi simboli sono indipendenti, e quindi si dimostra[18]



`https://www2.cs.duke.edu/csed/curious/compression/adaptivehuff.html`, mentre per una descrizione dell'algoritmo di VITTER `http://en.wikipedia.org/wiki/Adaptive_Huffman_coding`

[16] Infatti se calcoliamo $K = \sum_{k=1}^{L} 2^{-N_k}$ per N_k pari ai due valori indicati in (9.14) otteniamo nel primo caso

$$\sum_{k=1}^{L} 2^{-\log_2 \frac{1}{p_k}} = \sum_{k=1}^{L} 2^{\log_2 p_k} = \sum_{k=1}^{L} p_k = 1$$

mentre nel secondo

$$\sum_{k=1}^{L} 2^{-\left(\log_2 \frac{1}{p_k} + 1\right)} = \sum_{k=1}^{L} 2^{\log_2 p_k} \cdot 2^{-1} = 0.5 \sum_{k=1}^{L} p_k = 0.5$$

Pertanto in entrambi i casi la disuguaglianza di Kraft $K \leq 1$ è soddisfatta, e per valori intermedi si ottengono valori intermedi.

[17] Pari al numero di *disposizioni con ripetizione* di n oggetti estratti dagli elementi di un insieme di cardinalità L. Ad esempio, raggruppando due ($n = 2$) cifre decimali ($L = 10$), si ottiene un numero da 0 a 99, ovvero un simbolo ad $L^2 = 100$ valori.

[18] Indicando con y la v.a. aleatoria discreta in uscita dalla sorgente a blocchi, essa risulta di tipo *multivariato* (§ 6.2.6), le cui le v.a. marginali sono i simboli x emessi dalla sorgente originale. L'indipendenza statistica di questi ultimi consente di scrivere $Pr\{y\} = Pr\{x_1\}\, Pr\{x_2\} \cdots Pr\{x_n\}$ in cui i valori x_i sono quelli dei simboli *originali* che compongono y. L'entropia H_s^{blocco} della sorgente a blocchi è definita come valore atteso dell'informazione $I(y) = -\log_2 Pr(y)$, e in base alla proprietà del logaritmo di un prodotto possiamo scrivere $\log_2 Pr(y) = \log_2 Pr(x_1) + \log_2 Pr(x_2) + \cdots + \log_2 Pr(x_n)$, ottenendo cioè che $I(y)$ è pari alla somma dell'informazione legata ad ogni valore x che partecipa a comporre y. Pertanto si ottiene

$$\begin{aligned} H_s^{blocco} &= E\{I(y)\} = E\left\{-\log_2 Pr(x_1) - \log_2 Pr(x_2) - \cdots - \log_2 Pr(x_n)\right\} = \\ &= \sum_{j=1}^{n} E\left\{-\log_2 Pr(x)\right\} = nH_s \end{aligned}$$

ovvero l'entropia della sorgente equivalente è esattamente pari alla somma di quella dei simboli che

che l'entropia della nuova sorgente è n volte quella originale, ossia $H_s^{blocco} = nH_s$. Il risultato (9.15) quindi ora si scrive come $nH_s \leq n\bar{N} \leq nH_s + 1$ in cui $n\bar{N}$ è il numero medio di binit *per blocco*, e dividendo per n, otteniamo infine

$$H_s \leq \bar{N} \leq H_s + \tfrac{1}{n} \tag{9.16}$$

che rappresenta una forma di dimostrazione del teorema (9.10) con $\varepsilon = \frac{1}{n}$, e che permette di ottenere $\bar{N} \to H_s$ se $n \to \infty$, avvicinandosi alle condizioni di codifica ottima per qualsiasi distribuzione delle p_k.

Esercizio Per applicare questo metodo ad un caso pratico, consideriamo una sorgente *binaria* senza memoria che emette simboli x_k con probabilità p_k mostrata in tabella. Per ogni coppia di simboli il blocco serie/parallelo della fig. precedente emette un simbolo quaternario y_h a cui (in virtù dell'indipendenza statistica tra i simboli x_k) compete le probabilità ottenuta moltiplicando le probabilità della coppia $x_i x_j$ associata. Codifichiamo quindi i simboli y_h con il codice a lunghezza variabile introdotto al § 9.1.3, e ricalcoliamo il numero medio di binit/simbolo $\overline{N}$, che risulta pari a

Simbolo	Prob.	Codeword
x_1	0.8	1
x_2	0.2	0
$x_1x_1 \to y_1$	0.64	0
$x_1x_2 \to y_2$	0.16	10
$x_2x_1 \to y_3$	0.16	110
$x_2x_2 \to y_4$	0.04	111

$$\overline{N} = 1 \cdot 0.64 + 2 \cdot 0.16 + 3 \cdot 0.16 + 3 \cdot 0.04 = 1.58 \text{ binit}$$

ogni 2 simboli, ossia pari ad una media di 0.79 binit/simbolo binario, mentre in assenza di codifica a blocchi non avremmo potuto utilizzare meno di 1 binit/simbolo binario. Al crescere della dimensione di blocco n si può verificare come $\overline{N}$ si avvicini sempre più al valore dell'entropia della sorgente binaria $H_b = 0.72$ calcolato a pag. 253, ovvero dimostrare l'eq. (9.16).

9.1.4.1 Compromesso velocità-ritardo

Come indicato dalla (9.16), realizzando blocchi via via più lunghi è possibile ridurre la *velocità* media di codifica $\overline{N} \cdot f_s$ (in binit/sec) rendendo $\overline{N}$ sempre più vicino all'entropia, ovvero

$$\min\left[\bar{N}\right] = H_s + \varepsilon$$

in cui $\varepsilon \to 0$ se la lunghezza n del blocco tende ad infinito. D'altra parte, all'aumentare della dimensione del blocco aumenta di egual misura il *ritardo* che intercorre tra l'emissione di un simbolo e la sua codifica, e di questo va tenuto conto, nel caso sussistano dei vincoli temporali particolarmente stringenti sulla consegna del messaggio.

Riassumendo Qualora una sorgente discreta ad L simboli esibisca un valore di entropia inferiore a $\log_2 L$, la velocità binaria $\overline{N} \cdot f_s$ in uscita dal codificatore di sorgente può essere ridotta e resa prossima all'intensità informativa R (eq. (9.8)) adottando una codifica a blocchi di lunghezza via via crescente, e utilizzando per i nuovi simboli compositi un opportuno codice di Huffman.

rappresenta.

Esercizio Sperimentare la costruzione di un codice di Huffman basato sul raggruppamento di *tre* simboli della sorgente binaria dell'esercizio precedente, e verificare se il numero medio di binit/simbolo binario $\bar{N}$ riesce ad avvicinarsi ancora di più al valore dell'entropia della sorgente binaria pari a 0.72 bit/binit.

9.2 Sorgente discreta con memoria

Passiamo ora ad affrontare il caso in cui i simboli emessi dalla sorgente non possano essere ritenuti statisticamente indipendenti. Indicando con $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ una sequenza di n di simboli, la sua probabilità *congiunta* si calcola ora come

$$p(\mathbf{x}) = p(x_1)\, p(x_2/x_1)\, p(x_3/x_1, x_2) \ldots p(x_n/x_1, x_2, \ldots x_{n-1}) \neq \prod_{k=1}^{n} p(x_k) \qquad (9.17)$$

dato che appunto la dipendenza statistica comporta l'uso delle probabilità condizionali. L'espressione dell'entropia si modifica dunque in

$$H_n = E_{\mathbf{x}}\{I(\mathbf{x})\} = -\frac{1}{n} \sum_{\textit{tutte le possibili sequenze } \mathbf{x}} p(\mathbf{x}) \log_2 p(\mathbf{x}) \text{ bit/simbolo}$$

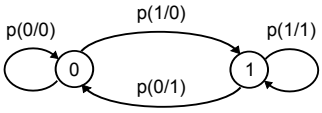
in cui $p(\mathbf{x})$ è la probabilità congiunta (9.17) di una possibile sequenza di simboli $\mathbf{x}$, e la media di insieme è effettuata su tutte le possibili sequenze $\mathbf{x}$ di lunghezza n. La grandezza H_n prende il nome di *entropia a blocco*, e si dimostra che al crescere di n il suo valore è *non crescente*, ossia $H_{n+1} \leq H_n \leq H_{n-1}$, mentre per $n \to \infty$, H_n tende ad un valore $H_\infty \leq H_s$, in cui l'uguaglianza è valida solo per sorgenti senza memoria.

9.2.1 Sorgente Markoviana

Se oltre ad un certo valore $n_{Max} = M$ la sequenza H_n non decresce più la sorgente è detta a *memoria finita* o *di Markov* di ordine M, ed è caratterizzata dal fatto che le probabilità condizionate dipendono solo dagli ultimi M simboli emessi.

Esempio Analizziamo il caso di una sorgente binaria di Markov del primo ordine, per la quale sono definite le probabilità condizionate mostrate a lato, a cui corrisponde il *diagramma di transizione* raffigurato. Essendo $M = 1$, lo *stato* della sorgente è determinato dal simbolo emesso per ultimo, che condiziona le probabilità di emissione del simbolo successivo: con i valori dell'esempio, si osserva come la sorgente *preferisca* continuare ad emettere l'ultimo simbolo prodotto, piuttosto che l'altro.

$$p(0/0) = 0.9 \quad p(1/0) = 0.1$$
$$p(0/1) = 0.4 \quad p(1/1) = 0.6$$



In pratica è come se ora vi fossero L^M diverse sorgenti S_i (nel caso dell'esempio, $2^1 = 2$), ognuna associata ad una diversa *storia passata* rappresentata dagli ultimi M simboli emessi (nell'esempio $M = 1$), identificativi dello *stato,* o *memoria,* della sorgente. In questo caso l'entropia di sorgente può essere calcolata applicando la (9.6) ad ognuno dei possibili stati, ottenendo in tal modo dei valori di *entropia condizionata* $H(x/S_i)$, mentre l'entropia di sorgente *complessiva* si ottiene come *valore atteso* dell'entropia

condizionata rispetto alle probabilità di trovarsi in ognuno degli stati del modello Markoviano.

Tornando all'esempio, i valori di entropia condizionata risultano pari a

$$\begin{aligned} H(x/S_0) &= -0.9\log_2 0.9 - 0.1\log_2 0.1 = 0.47 \\ H(x/S_1) &= -0.4\log_2 0.4 - 0.6\log_2 0.6 = 0.97 \end{aligned}$$

bit/simbolo, mentre il valore della probabilità di trovarsi in uno dei due stati si ottiene risolvendo il sistema

$$\begin{cases} p(S_0) &= p(0/0)\,p(S_0) + p(0/1)\,p(S_1) \\ 1 &= p(S_0) + p(S_1) \end{cases} \tag{9.18}$$

in cui la prima equazione asserisce che la probabilità di trovarsi in S_0 è pari alla somma di quella di esserci già, per quella di emettere ancora zero, più la probabilità di aver emesso uno, ed ora emettere zero. Procedendo per sostituzione si ottiene $p(S_0) = 0.8$ e $p(S_1) = 0.2$, ossia gli stessi valori dell'esempio binario senza memoria di pag. 253. Ma mentre in quel caso il valore dell'entropia risultava pari a 0.72 bit/simbolo, ora si ottiene

$$H = p(S_0)\,H(x/S_0) + p(S_1)\,H(x/S_1) = 0.58 \text{ bit/simbolo}$$

mostrando come la presenza di memoria aumenti la predicibilità delle sequenze emesse dalla sorgente.

Esercizio Si ripeta il calcolo dell'entropia per un modello di Markov del primo ordine, caratterizzato dalle probabilità $p(0) = p(1) = 0.5$ e $p(1/0) = p(0/1) = 0.01$, mostrando che in questo caso si ottiene una entropia di 0.08 bit/simbolo.

9.2.1.1 Autovettore della matrice di transizione

Come qualcuno avrà notato, le probabilità $p(S_0)$ e $p(S_1)$ corrispondono alla statistica di ordine zero (ossia la probabilità incondizionata, o marginale) dei simboli di sorgente $x_0 = 0$ ed $x_1 = 1$. I valori $p(S_i)$ possono essere ottenuti (anziché impostando il sistema di equazioni di cui all'esempio precedente) come gli elementi dell'autovettore $\boldsymbol{v}_\pi$ associato all'autovalore 1 della *matrice di transizione* Π, i cui elementi $\pi_{i,j}$ sono pari alle probabilità condizionate $p(x_i/x_j)$, potendo scrivere $\Pi \cdot v_\pi = v_\pi$. In pratica ogni riga di tale sistema di equazioni è l'estensione della prima delle (9.18) a ciascuno stato della sorgente markoviana.

9.2.2 Codifica per sorgenti con memoria

La discussione svolta fin qui mostra come l'entropia delle sorgenti con memoria sia sempre *minore* di quella relativa al caso di indipendenza statistica dei simboli emessi, e qualora le probabilità condizionate siano note al codificatore, possono essere usate per calcolare le probabilità per blocco (9.17) e con queste individuare un codice di Huffman in grado di ridurre anche la velocità di codifica, a patto di accettare un maggior ritardo legato all'uso di codici a blocchi.

Ma la dimensione dei blocchi da prendere in considerazione può divenire eccessiva, producendo spropositate tabelle di codeword. Inoltre si può ritenere di *non* conoscere

la statistica della sorgente, e di non desiderare effettuarne una stima, ed evitare di trasmettere la tabella di codeword. In questi casi, può essere opportuno adottare tecniche diverse dalle precedenti, come le due riportate appresso.

9.2.2.1 Codifica run-length

Prendendo come esempio tipico il caso della trasmissione fax, in cui si ha a che fare con un segnale di immagine in bianco e nero, scansionato per righe, che è assimilabile ad una sorgente binaria che emetta uno zero per il bianco, ed un uno per il nero: per la natura delle immagini scansionate, tipicamente ci saranno lunghe sequenze di uni o di zeri, e dunque si può assumere valido un modello di sorgente Markoviano di primo ordine, con elevata probabilità condizionata di rimanere nello stesso stato.

Le lunghe sequenze di bit tutti uguali vengono dette *run* (corse), e la codifica *run-length* consiste nel trasmettere parole di codice che indicano il numero (*length*) di questi bit uguali. In questo caso quindi la codeword è di lunghezza fissa (ad esempio $n+1$ binit, il primo dei quali indica se il run è tutto di uni o di zeri), e rappresenta un numero variabile (da 0 a 2^n-1) di binit di sorgente. Se ad esempio $n=6$ binit, questi 6+1 = 7 binit possono codificare fino a 64 (uguali) simboli di sorgente binaria: un bel risparmio![19]

9.2.2.2 Codifica predittiva

Questa ulteriore tecnica si basa sul fatto che un elevato grado di dipendenza statistica dei messaggi comporta la possibilità di *predire* in qualche modo i simboli a venire, in base all'identità di quelli già emessi. La differenza tra la sequenza predetta $\hat{x}$ e quella effettiva x è una nuova sequenza indicata come *errore di predizione* $e=\hat{x}-x$ che, se il predittore *ci azzecca* per la maggior parte del tempo, è quasi tutta nulla. Il predittore conserva uno *stato interno* che rappresenta gli ultimi bit di ingresso, in base ai quali determina[20] la stima $\hat{x}$, da cui ottenere l'errore e (frequentemente nullo) che viene sottoposto a codifica run-length, e trasmesso.

Ora avviene *un trucco*, dato che in realtà il predittore *non conosce* la sequenza $x_{k-1}, \cdots x_{k-M}$ da cui predire $\hat{x}_k$, ma viene invece *alimentato* con la sequenza di errore e, in modo che da questa possa determinare l'ultimo (vero) simbolo di ingresso come $x=\hat{x}-e$, ed aggiornare il proprio stato interno. Dal lato ricevente opera un predittore identico a quello di codifica, anch'esso alimentato dalla sequenza e in modo da generare un valore $\hat{x}$ identico a quello del codificatore, a cui si applica la stessa relazione $x=\hat{x}-e$

[19] In realtà, nel caso specifico del fax le cose non stanno esattamente in questi termini: infatti, anziché usare una parola di lunghezza fissa di n binit, l'ITU-T ha definito un apposito codebook `http://www.itu.int/rec/T-REC-T.4-199904-S/en` che rappresenta un codice di Huffman a lunghezza variabile, in modo da codificare le run length più frequenti con un numero ridotto di bit.

[20] Il lettore più curioso si chiederà a questo punto, come è fatto il predittore. Molto semplicemente, *scommette* sul prossimo simbolo più probabile, in base alla conoscenza di quelli osservati per ultimi, ed ai parametri del modello markoviano: se il prossimo simbolo viene predetto in base ad una sua probabilità condizionata > 0.5, allora *la maggior parte* delle volte la predizione sarà corretta, ed il metodo consegue una riduzione di velocità. Nel caso di sorgenti continue, al § 9.5.4 troveremo invece alcune particolarità aggiuntive.

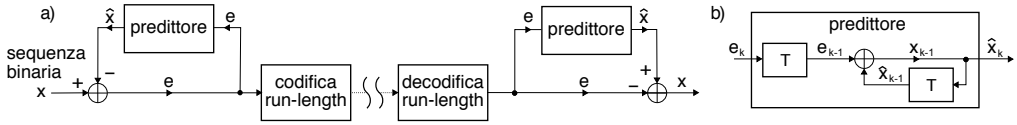


Figura 9.2: a) - schema di un codificatore predittivo binario; b) - architettura del predittore del primo ordine

per ottenere il valore del simbolo x correttamente *decodificato*. Perché lo schema possa funzionare, occorre che i due predittori condividano il medesimo stato interno iniziale.

La fig. 9.2-a) esemplifica l'applicazione della tecnica al caso di sequenze binarie, per le quali l'operazione di differenza è realizzata tramite una somma modulo due $\oplus$, dato che scrivendo $e = \hat{x} \oplus x$ risulta $e \neq 0$ solamente quando $\hat{x} \neq x$. La parte destra della figura mostra quindi l'architettura interna del predittore del primo ordine adottato, che cioè tiene conto del solo simbolo *precedente* x_{k-1}, e che semplicemente *scommette* che sia uguale al successivo, ovvero $\hat{x}_k = x_{k-1}$; il *vero* simbolo precedente x_{k-1} è ottenuto come prima descritto, ovvero a partire dalla stima fornita in precedenza $\hat{x}_{k-1}$ ottenuta mediante un ritardo, e sommata al precedente errore e_{k-1}, anch'esso ritardato rispetto alla sequenza di ingresso.

Notiamo come la codifica run-length non preveda l'esistenza di un accordo a priori tra trasmettitore e ricevitore, a parte il comune stato di partenza ad inizio messaggio, mentre la codifica predittiva necessita solo di un accordo in merito alla struttura del predittore. Per contro, in presenza di errori di trasmissione i due predittori restano disallineati, finché non si inizia a co-decodificare un nuovo messaggio. Ma lo stesso problema è comune anche al caso di codifica a lunghezza di parola variabile, ed a quello di Huffman dinamico.

9.2.3 Compressione basata su dizionario

Nella comune accezione del termine un dizionario è costituito da un *array* di stringhe, popolato con le parole esistenti per un determinato linguaggio. Anziché operare carattere per carattere, un codificatore di sorgente *testuale* potrebbe ricercare la posizione nel dizionario delle parole del messaggio, e quindi trasmettere l'indice della parola: per un dizionario di 25.000 termini bastano 15 bit di indirizzamento, ottenendo un rapporto di compressione variabile, in funzione della lunghezza della parola codificata.

9.2.3.1 Metodo di Lempel-Ziv-Welsh

Per evitare di dover condividere la conoscenza dell'intero dizionario tra sorgente e destinatario, che tra l'altro potrebbe essere assolutamente sovradimensionato rispetto alle caratteristiche dei messaggi da trattare, il metodo LZW prevede che il codificatore generi il dizionario in modo graduale, man mano che analizza il testo, e che il decodificatore sia in grado di replicare questa modalità di generazione. Inoltre, il dizionario non è vincolato a contenere le reali *parole* del messaggio, ma semplicemente ospita le sequenze di caratteri effettivamente osservati, di lunghezza due, tre, quattro...

Operando su di un alfabeto ad L simboli, rappresentabili con $n = \lceil \log_2 L \rceil$ bit, il dizionario iniziale conterrà i simboli di sorgente alle prime L posizioni, e posti liberi

nelle restanti $2^n - L$ posizioni[21]. Ogni carattere c letto in ingresso viene accodato in una *stringa* w, ed il risultato confrontato con le stringhe già presenti nel dizionario. Nel caso non si verifichi nessuna corrispondenza, viene aggiunta una nuova voce di dizionario, e quindi viene trasmesso l'indice associato alla sua parte iniziale, escludendo cioè il simbolo concatenato per ultimo, e che ha prodotto l'occorrenza della nuova voce. Nel caso invece in cui la stringa sia già presente (e questo in particolare è vero per la stringa di lunghezza uno corrispondente al primo simbolo analizzato) non si emette nulla, ma si continuano a concatenare simboli fino ad incontrare una stringa mai vista. Presso *Wikipedia*[22] è presente un esempio di risultato della codifica.

```
w = NIL;
while (read a char c) do
  if (wc exists in dictionary) then
    w = wc;
  else
    add wc to the dictionary;
    output the code for w;
    w = c;
  endif
done
output the code for w;
```

La parte iniziale del testo, ovviamente, ha una alta probabilità di contenere tutte coppie di caratteri mai viste prima, e quindi in questa fase vengono semplicemente emessi i codici associati ai simboli osservati. Con il progredire della scansione, aumenta la probabilità di incontrare stringhe già osservate e sempre più lunghe. Ogni volta che viene esaurito lo spazio residuo per i nuovi simboli, viene aggiunto un bit alla lunghezza della codeword, ovvero viene raddoppiata la dimensione del vocabolario. Man mano che viene analizzato nuovo materiale, aumenta la lunghezza delle stringhe memorizzate nel dizionario, che riflette l'effettiva composizione statistica del documento in esame, ivi compresa la presenza di memoria (nel senso di dipendenza statistica); allo stesso tempo, la dimensione del dizionario (e la lunghezza delle codeword) resta sempre la minima indispensabile per descrivere il lessico effettivamente in uso. Alla fine del processo, il dizionario ottenuto può essere aggiunto[23] *in testa* al file compresso, seguito dalle codeword risultanti dall'algoritmo.

L'algoritmo LZW è usato nel programma di compressione Unix `compress`, per la realizzazione di immagini GIF e TIFF, ed incorporato in altri software, come ad esempio *Adobe Acrobat*.

9.2.3.2 Algoritmo Deflate

L'ultimo metodo di compressione senza perdite che esaminiamo è quello che è stato introdotto da PHIL KATZ[24] con il programma PKZIP, e quindi formalizzato nella RFC 1951[25], e tuttora ampiamente utilizzato per le sue ottime prestazioni e l'assenza di

[21] Ad esempio con $L = 96$ simboli si ha $n = 7$, ed un dizionario iniziale con 128 posizioni, di cui 96 occupate e 32 libere.

[22] Vedi ad es. `http://en.wikipedia.org/wiki/Lempel-Ziv-Welch`

[23] Il realtà il dizionario *non viene* aggiunto, ma *ri-generato* durante il processo di decodifica, come illustrato al link di cui alla nota precedente.

[24] Vedi ad es. `https://en.wikipedia.org/wiki/Phil_Katz`

[25] Vedi ad es. `http://tools.ietf.org/html/rfc1951`

brevetti. Usa una variante dell'algoritmo LZW, al cui risultato applica poi una codifica di Huffman. *Deflate* opera su blocchi di dati con dimensione massima 64 Kbyte, ognuno dei quali può essere replicato intatto (come nel caso in cui i bit siano già sufficientemente impredicibili), oppure essere compresso con un codice di Huffman statico, oppure ancora dinamico.

Per quanto riguarda la variante di LZW, essa consiste nel *non costruire* esplicitamente il dizionario, ma nell'usare invece *dei puntatori all'indietro* per specificare che una determinata sotto-stringa di ingresso, è in realtà la ripetizione di un'altra già osservata in precedenza. In questo caso, anziché emettere il codice (di Huffman) associato alla codeword già presente nel dizionario, si emette (il codice di Huffman del) la lunghezza della stringa da copiare, e la distanza (nel passato) della stessa. Quindi in pratica, anziché usare una codeword di lunghezza fissa per indicizzare gli elementi del dizionario come per LZW, viene usato un puntatore di lunghezza variabile, privilegiando le copie della sottostringa corrente più prossime nel tempo, oppure quelle con un maggior numero di caratteri uguali.

9.3 Contenuto informativo di sorgente continua

Sebbene l'estensione del concetto di entropia già definito per sorgenti discrete (§ 9.1.1) sia abbastanza diretto, la sua applicazione al caso di sorgenti tempo-continue presenta risvolti particolari, che andiamo a discutere.

9.3.1 Entropia differenziale di sorgente continua

L'espressione (9.2) valida per le sorgenti discrete può essere formalmente estesa al caso di una sorgente continua che produce un processo $x(t)$ stazionario ed incorrelato, descritto da una d.d.p. del primo ordine $p_x(x)$, portando all'espressione

$$h(X) = E\left\{-\log_2 p_x(x)\right\} = -\int p_x(x) \log_2 p_x(x)\, dx \tag{9.19}$$

indicata con la h minuscola per distinguerla dal caso discreto, e chiamata *entropia differenziale* a seguito delle proprietà che andiamo ad illustrare.

Dipendenza dalla dinamica Il valore ottenuto dalla (9.19) può risultare positivo, negativo o nullo, in funzione della dinamica della variabile aleatoria X.

Esempio Se calcoliamo il valore di entropia differenziale per un processo i cui valori sono descritti da una variabile aleatoria a distribuzione uniforme $p_x(x) = \frac{1}{A} rect_A(x)$, otteniamo il risultato $h(X) = -\frac{1}{A}\int_{-A/2}^{A/2} \log_2\left(\frac{1}{A}\right) dx = \log_2 A$ il cui valore effettivo, appunto, dipende dal valore di A. In particolare, se $A = 1$ si ottiene $h(X) = 0$.

L'esempio è un modo per osservare che, in presenza di un fattore di una v.a. $Y = \alpha X$ *scalata* di un fattore α, si ottenga $h(Y) = h(X) + \log_2 |\alpha|$.

Invarianza rispetto alla media L'entropia differenziale non dipende dal valor medio della variabile aleatoria, ovvero è invariante rispetto alle traslazioni. Per verificare

la veridicità di tale affermazione, calcolare per esercizio il valore di $h(X)$ per una d.d.p. $p_x(x) = \frac{1}{A} rect_A(x - m)$.

Confronto tra entropia di processi Essendo il valore $h(X)$ dipendente dalla dinamica della v.a., l'entropia differenziale sembra inadatta ad esprimere il contenuto informativo *assoluto* di una sorgente continua; ciononostante può comunque essere utile per confrontare due sorgenti con *uguale varianza* σ_x^2, come mostrato alla nota[26]. A tale proposito, valutiamo il valore di entropia differenziale per un caso particolarmente rilevante.

9.3.2 Entropia differenziale di sorgente gaussiana

Applicando la (9.19) al caso $p_X(x) = \frac{1}{\sqrt{2\pi}\sigma_x} e^{-\frac{x^2}{2\sigma_x^2}}$, dopo aver osservato (vedi nota 1) che

$$-\log_2 p(x) = -\frac{\ln p(x)}{\ln 2} = \frac{1}{\ln 2}\left(\ln\sqrt{2\pi\sigma_x^2} + \frac{x^2}{2\sigma_x^2}\right)$$

possiamo scrivere

$$\begin{aligned} h_G(X) &= -\int p(x)\log_2 \frac{1}{\sqrt{2\pi}\sigma_x} e^{-\frac{x^2}{2\sigma_x^2}} dx = \int p(x)\frac{1}{\ln 2}\left(\ln\sqrt{2\pi\sigma_x^2} + \frac{x^2}{2\sigma_x^2}\right)dx = \\ &= \frac{1}{\ln 2}\left(\ln\sqrt{2\pi\sigma_x^2}\int_{-\infty}^{\infty} p(x)\,dx + \frac{1}{2\sigma_x^2}\int_{-\infty}^{\infty} p(x)\,x^2 dx\right) = \\ &= \frac{1}{\ln 2}\left(\ln\sqrt{2\pi\sigma_x^2} + \frac{1}{2}\right) = \frac{1}{\ln 2}\ln\sqrt{2\pi e\sigma_x^2} = \log_2\sqrt{2\pi e\sigma_x^2} = \\ &= \frac{1}{2}\log_2\left(2\pi e\sigma_x^2\right) \end{aligned} \tag{9.20}$$

essendo $\frac{1}{2} = \ln e^{1/2}$, ed avendo di nuovo applicato la nota 1.

9.3.2.1 Massima informazione per processo gaussiano

Al § 9.6.2 si mostra che il processo gaussiano è quello che consegue il massimo valore di entropia differenziale per σ_x^2 assegnata, ovvero è valida la diseguaglianza

$$h_G(X) = \frac{1}{2}\log_2\left(2\pi e\sigma_x^2\right) > h(X) \quad \text{data } \sigma_x^2 \tag{9.21}$$

[26] In effetti esiste una misura di entropia *assoluta* per sorgenti continue, che però ha la *sgradevole caratteristica* di risultare sempre infinita. Infatti, approssimando la (9.19) come limite a cui tende una sommatoria, e suddividendo l'escursione dei valori di x in intervalli uguali Δx, possiamo scrivere

$$\begin{aligned} h_{abs}(x) &= \lim_{\Delta x\to 0}\sum_i p(x_i)\,\Delta x \log_2\frac{1}{p(x_i)\,\Delta x} \\ &= \lim_{\Delta x\to 0}\sum_i\left[p(x_i)\,\Delta x\log_2\frac{1}{p(x_i)} + p(x_i)\,\Delta x\log_2\frac{1}{\Delta x}\right] = h(x) + h_0 \end{aligned}$$

in cui $h(x)$ è proprio la (9.19) mentre $h_0 = -\lim_{\Delta x\to 0}\log_2\Delta x\int_{-\infty}^{\infty} p(x)\,dx = -\lim_{\Delta x\to 0}\log_2\Delta x = \infty$. D'altra parte, la differenza tra le entropie assolute di due sorgenti z e x risulta pari a $h_{abs}(z) - h_{abs}(x) = h(z) - h(x) + h_0(z) - h_0(x)$, in cui la seconda differenza tende a $-\log_2\frac{\Delta z}{\Delta x}$ che, se z ed x hanno la medesima dinamica, risulta pari a zero.

Principio di massima entropia In presenza di informazioni incomplete a riguardo di un sistema stocastico, come ad es. la conoscenza della sola varianza di una v.a., assumere l'ipotesi di gaussianità a riguardo della d.d.p. che lo governa equivale[27] ad adottare *le ipotesi meno restrittive* (ovvero più informative) a riguardo..

9.4 Misure di informazione per una coppia di v.a.

Descrivono da un punto di vista informativo i messaggi prodotti da una coppia di sorgenti, ovvero componenti di una v.a. bidimensionale; i risultati ottenuti saranno utilizzati nel contesto della quantizzazione (§ 9.5) e della codifica di canale (cap. 17). Vengono poi introdotti altri due risultati (§§ 9.4.4 e 9.4.5), utilizzati specificatamente in altri contesti. Per semplicità, le definizioni vengono espresse nei termini di v.a. discrete.

9.4.1 Entropia congiunta

Si riferisce a due v.a. X e Y le cui realizzazioni sono descritte dalle d.d.p. marginali $p(x)$ e $p(y)$ e dalla d.d.p. congiunta $p(x,y)$, ed è definita come

$$H(X,Y) = H(Y,X) = -\sum_x \sum_y p(x,y) \log_2 p(x,y)$$

L'entropia congiunta risulta sempre non negativa, e delimitata tra

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

con l'ultimo $\leq$ che diviene un'uguaglianza qualora le v.a. siano statisticamente indipendenti ovvero $p(x,y) = p(x)\,p(y)$. Nel caso di v.a. continua sussiste l'equivalente definizione per l'entropia differenziale congiunta

$$h(X,Y) = -\int_x \int_y p(x,y) \log_2 p(x,y)\, dxdy \tag{9.22}$$

9.4.2 Entropia condizionale

Come la precedente si riferisce a due v.a. X e Y descritte dalle d.d.p. $p(x)$, $p(y)$ e dalla d.d.p. condizionata $p(y/x) = {}^{p(x,y)}/_{p(x)}$, viene definita come

$$H(Y/X) = -\sum_x \sum_y p(x,y) \log_2 p(y/x) \tag{9.23}$$

e per essa sussiste la relazione

$$H(Y/X) = H(X,Y) - H(X) \tag{9.24}$$

che si ottiene dalla (9.23) considerando che $p(y/x) = {}^{p(x,y)}/_{p(x)}$. In base alla relazione analoga per $p(x/y)$ è altrettanto vero che $H(X/Y) = H(X,Y) - H(Y)$, e dunque sussiste anche *l'equivalente* del teorema di Bayes (§ 6.1.4), ovvero $H(Y/X) = H(X/Y) + H(Y) - H(X)$.

La (9.24) può essere *interpretata* considerando che mentre $H(X,Y)$ esprime il numero medio di bit di informazione associati alla conoscenza di una coppia di realizzazioni (x,y), l'osservazione della sola v.a. X apporta una informazione media di $H(X)$ bit/simbolo. Pertanto sono necessari solamente $H(X,Y) - H(X)$ ulteriori bit (in media) per descrivere anche la conoscenza di Y, una volta che X sia nota. Per la

[27] Approfondimenti presso `https://en.wikipedia.org/wiki/Principle_of_maximum_entropy`

(9.23) risulta

$$0 \leq H(Y/X) \leq H(Y)$$

in cui la prima relazione è una uguaglianza se (e solo se) $p(y/x)$ è una funzione deterministica e non una d.d.p., mentre $H(Y/X) = H(Y)$ se (e solo se) $p(y,x) = p(x)\,p(y)$ e quindi $p(y/x) = p(y)$.

Nel caso di v.a. continue la definizione di entropia differenziale condizionale è

$$h(Y/X) = -\int_x \int_y p(x,y) \log_2 p(y/x)\, dxdy = -\int_x p(x) \int_y p(y/x) \log_2 p(y/x)\, dydx$$

i cui valori possono però risultare anche negativi o indeterminati (pag. 266).

9.4.3 Informazione mutua media

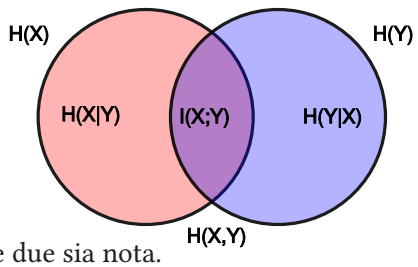
Anche questa grandezza tiene conto di due v.a. X e Y[28] descritte dalle d.d.p. marginali $p(x)$ e $p(y)$, nonché dalla d.d.p. congiunta $p(x,y)$; la sua definizione è

$$I(X;Y) = I(Y;X) = \sum_x \sum_y p(x,y) \log_2 \frac{p(x,y)}{p(x)\,p(y)} \tag{9.25}$$

ed ha un valore positivo o nullo, quest'ultimo se (e solo se) la v.a. sono indipendenti, ovvero $p(x,y) = p(x)\,p(y)$. Il valore di $I(X,Y)$ misura l'informazione che X e Y *condividono*, ovvero quanto la conoscenza di una riduce l'incertezza a riguardo dell'altra. Per essa sussistono le eguaglianze[29]

$$\begin{aligned} I(X;Y) &= H(X) - H(X/Y) = H(Y) - H(Y/X) \\ &= H(X) + H(Y) - H(X,Y) = \\ &= H(X,Y) - H(X/Y) - H(Y/X) \end{aligned}$$

che possono essere meglio apprezzate nei termini di unione, differenza ed intersezione di insiemi, come raffigurato nel diagramma mostrato a lato. In particolare, in base alle prime due eguaglianze possiamo dire che $I(X;Y)$ è pari all'entropia di una delle due v.a., meno il numero di bit a simbolo necessari a descriverla qualora l'altra v.a. sia nota, ovvero meno l'*incertezza residua* qualora una delle due sia nota.



Anche questo concetto si applica al caso di v.a. continue, ottenendo l'espressione dell'informazione mutua media *differenziale*

$$I(X;Y) = \int_x \int_y p(x,y) \log_2 \frac{p(x,y)}{p(x)\,p(y)} dxdy$$

che *non dipende* dalla dinamica[30] delle v.a. X e Y come invece accadeva per l'entropia differenziale di una (9.19) o due (9.22) v.a.

[28] Vedi anche la trattazione al § 17.1.3 e seguenti nel caso in cui X ed Y siano le grandezze in ingresso ed in uscita da un canale di comunicazione.

[29] Vedi ad es. `https://en.wikipedia.org/wiki/Mutual_information`, ma anche la nota 5 a pag. 555

[30] Ciò deriva dall'essere le d.d.p. presenti sia a numeratore che a denominatore dell'argomento di $\log_2$.

9.4.4 Entropia relativa

Meglio nota come *Divergenza di Kullback Leibler*, è una misura di quanto una d.d.p. $p(x)$ è differente da una seconda $q(x)$, di riferimento. E' definita dall'espressione[31]

$$D_{KL}(p||q) = \sum_x p(x) \log_2 \frac{p(x)}{q(x)} = -\sum_x p(x) \log_2 \frac{q(x)}{p(x)} \tag{9.26}$$

ed è descritta anche come entropia relativa *da q a p*, o divergenza *di p da q*. Il suo valore è positivo o nullo, e si azzera quando $p(x) = q(x)$; non può essere però adottata come una *distanza*, in quanto non è simmetrica (ovvero $D_{KL}(p||q) \neq D_{KL}(q||p)$) e non verifica la disuguaglianza triangolare.

In genere $p(x)$ deriva da osservazioni sperimentali, mentre $q(x)$ ne rappresenta un modello teorico, ed il valore di $D_{KL}(p||q)$ può essere interpretato come il numero medio di bit *in più* necessario a codificare i simboli x adottando un codice ottimizzato rispetto a $q(x)$, anziché uno ottimizzato per $p(x)$. In questo senso, $D_{KL}(p||q)$ misura il *guadagno di informazione* conseguito nel rivedere le proprie convinzioni a riguardo del fenomeno aleatorio espresso dalla v.a. X, da una d.d.p. *a priori* $q(x)$, in favore della evidenza basata sui dati $p(x)$.

Nel caso di v.a. continue la definizione (9.26) si modifica in

$$D_{KL}(p||q) = \int_x p(x) \log_2 \frac{p(x)}{q(x)} dx$$

e, come per $I(X;Y)$, il suo valore non dipende dalla dinamica delle v.a.

Relazione con l'informazione mutua media La (9.26) può essere vista come il valore di informazione mutua media (9.25) tra due v.a. X e Y, calcolato come divergenza $D_{KL}(p(x,y) \,||\, p(x)\,p(y))$ della d.d.p. congiunta $p(x,y)$ dal prodotto $p(x)\,p(y)$ delle rispettive marginali. Dato però che la (9.25) prevede una doppia sommatoria mentre la (9.26) soltanto una, si preferisce scrivere

$$\begin{aligned} I(X;Y) &= \sum_x \sum_y p(x,y) \log_2 \frac{p(x,y)}{p(x)\,p(y)} = \sum_x \sum_y p(x/y)\,p(y) \log_2 \frac{p(x/y)\,p(y)}{p(x)\,p(y)} = \\ &= \sum_y p(y) \sum_x p(x/y) \log_2 \frac{p(x/y)}{p(x)} = E_Y \{D_{KL}(p(x/y) \,||\, p(x))\} \end{aligned}$$

in cui la somma su x valuta la divergenza della d.d.p. condizionata $p(x/y)$ dal modello $p(x)$, mentre la somma esterna su y esegue il valore atteso rispetto ai valori y. Più $p(x/y)$ e $p(x)$ sono differenti (o divergenti), e maggiore è il guadagno di informazione.

9.4.5 Entropia di Rényi

Definita in tempi più recenti, estende il concetto di entropia introdotto da Shannon (9.2), che ne diviene un caso particolare. L'entropia di Rényi di ordine α, con $\alpha \geq 0$ ed $\alpha \neq 1$, di una v.a. discreta X con alfabeto di n elementi x_i, di probabilità p_i, è definita come[32]

$$H_\alpha(X) = \frac{1}{1-\alpha} \log_2 \left(\sum_{i=1}^n p_i^\alpha \right)$$

[31] Vedi ad es. `https://en.wikipedia.org/wiki/Kullback-Leibler_divergence`

[32] Vedi ad es. `https://en.wikipedia.org/wiki/Rényi_entropy`

In caso di v.a. uniforme con $p_i = 1/n \ \forall i$ si ha $H_\alpha(X) = \log_2 n$ per $\forall \alpha$; in generale, $H_\alpha(X)$ è una funzione non crescente di α. Per diversi specifici valori di α accade che

- per $\alpha \to 0$ si ha $H_0(X) \to \log_2 n$ per qualunque d.d.p.;
- per $\alpha \to 1$ il valore $H_1(X)$ eguaglia quello dell'entropia *classica* $\sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i}$;
- per $\alpha \to \infty$ il valore $H_\alpha(X)$ è sempre più legato ai soli eventi più probabili.

Viene inoltre definita una *divergenza di Rényi* tra due d.d.p. discrete p_i e q_i di uguale cardinalità n come $D_\alpha(p||q) = \frac{1}{\alpha-1} \log_2 \left(\sum_{i=1}^{n} \frac{p_i^\alpha}{q_i^{\alpha-1}} \right)$ che, per $\alpha \to 1$, corrisponde alla divergenza di Kullback-Leibler (§ 9.4.4).

9.5 Codifica di sorgente con perdita di informazione

Si rende necessaria quando il *canale trasmissivo* non può trasportare un flusso informativo qualsiasi[33], ma esiste un limite R_M alla massima velocità di codifica R, ovvero si impone che $R \leq R_M$. Ciò è possibile a patto di accettare una *perdita di informazione* che si traduce nell'insorgere di una distorsione D del messaggio, di cui vogliamo stabilire l'entità $D(R)$, in funzione della velocità R. Siamo altresì interessati a stabilire il viceversa, ovvero quale sia la minima velocità di trasmissione $R(D)$, qualora si accetti una distorsione $D \leq D_M$.

Messaggi di natura discreta Quando la massima velocità R_M è inferiore al tasso informativo (9.8) della sorgente, anche la codifica più efficiente sviluppa una velocità R eccessiva e dunque si è obbligati a *scartare* informazione, come ad esempio omettere la trasmissione di intere codeword, con la conseguenza di introdurre errori (o distorsione D) nel messaggio codificato[34], in proporzione tanto maggiore quanto minore è R_M.

Sorgenti continue Qui abbiamo due strade possibili: o intraprendere un processo di campionamento e quantizzazione (§ 4.3) per produrre un segnale numerico con velocità $R \leq R_M$, ed incorrere in una distorsione di quantizzazione D tanto maggiore quanto minore è la R_M consentita, oppure effettuare una trasmissione analogica del segnale $x(t)$ con una potenza S, con le conseguenze

- di ricevere $r(t) = x(t) + n(t)$, in cui il rumore $n(t)$ di potenza N in ricezione causa una distorsione $d(t) = n(t) = x(t) - r(t)$ di potenza $D = N$;
- l'entità di tale distorsione D può essere ridotta aumentando il rapporto $\text{SNR} = S/N$ oppure la banda del canale W, ovvero aumentandone la capacità C[35] e di pari misura la massima intensità di informazione R_M che può essere trasferita.

[33] Questo limite può essere causato da una insufficiente capacita di canale (§ 17.3), o da una limitata disponibilità di risorse, come ad es. nella archiviazione dei dati in memoria.

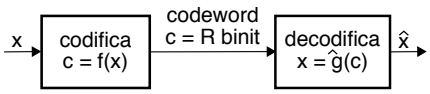
[34] La perdita di informazione per messaggi discreti determina la *corruzione* del messaggio, come la mancanza di parti di testo, di un'immagine, o di un video. Ma nel caso di *trasmissione dati* si preferisce *impiegare più tempo* per la trasmissione, piuttosto che perdere informazione.

[35] Al § 17.3 si mostra come la massima intensità di informazione R (associata ad un segnale $x(t)$ di potenza S ricevuto dopo aver attraversato un canale ideale con banda W ed alla cui uscita è presente un rumore $n(t)$ gaussiano bianco di potenza N) *non può superare* un limite C noto come *capacità di canale*, pari a $C = W \log_2 \left(1 + \frac{S}{N}\right)$ bit per secondo.

Nel caso continuo pertanto, sia che la sorgente venga quantizzata, sia che se ne effettui la trasmissione come segnale analogico, sussiste un legame inverso tra distorsione e velocità di trasmissione, il cui studio prende il nome di *teoria velocità-distorsione*, di cui discutiamo ora.

9.5.1 La distorsione di codifica

Per individuare la relazione $R(D)$ tra la minima velocità R di trasmissione di una codifica (con perdite) di una sorgente informativa e la distorsione D che si ritiene accettabile ci riferiamo alla figura a lato, in cui un simbolo (sorgente discreta) o valore (s. continua) x da trasmettere viene rappresentato mediante codeword $c = f(x)$ di R binit, limitando dunque l'alfabeto a 2^R elementi, a cui si associa in ricezione un nuovo valore $\hat{x} = g(f(x)) \neq x$. La corrispondenza tra x ed $\hat{x}$ viene quindi espressa in termini probabilistici come $p(\hat{x}/x)$, mentre indichiamo con $d(x, \hat{x})$ la misura della distorsione corrispondente[36]. A questo punto è possibile definire la *distorsione media* D_x in cui si incorre nel rappresentare un generico valore (o simbolo) x mediante $\hat{x}$ come un valore atteso, ossia



$$D_x = E_{\hat{X},X}\{d(\hat{x}, x)\} = \iint p(\hat{x}/x)\, p(x)\, d(\hat{x}, x)\, d\hat{x}dx$$

la cui entità dipende sia dalla scelta della misura $d(x, \hat{x})$ che dalla d.d.p. $p(\hat{x}/x)$, risultato della scelta delle operazioni di codifica $c = f(x)$ e restituzione $\hat{x} = g(c)$.

9.5.2 Funzione velocità-distorsione

E' il nome dato alla relazione $R(D)$ definita come la soluzione ad un problema di *minimizzazione*[37], ossia come quella che rende minima l'informazione mutua media $I(\hat{x}; x)$ tra x e $\hat{x}$ al variare di $p(\hat{x}/x)$ in tutti i modi possibili, in modo da mantenere D_x inferiore alla distorsione desiderata D:

$$R(D) = \min_{p(\hat{x}/x):\, D_x \leq D} I(\hat{X}; X) \tag{9.27}$$

Ricordando le osservazioni svolte al § 9.4.3 a proposito di $I(\hat{X}; X)$, questa misura quanto la conoscenza di $\hat{x}$ riduce l'incertezza a riguardo di x, incertezza che altrimenti[38] sarebbe pari all'entropia $H(X)$: intuitivamente si desidera che tale riduzione sia la massima possibile, e dunque la minimizzazione espressa dalla (9.27) va considerata

[36] Per sorgenti continue $d(\hat{x}, x)$ può ad es. corrispondere ad un valore quadratico medio (o varianza, o potenza) dell'errore di quantizzazione dei suoi campioni (§ 4.1), mentre $p(\hat{x}/x)$ dipende dalla caratteristica di quantizzazione $f(x)$ (§ 4.3), e fornisce prob. pari ad uno al valore quantizzato $\hat{x}_k$ (o *centroide*) associato all'intervallo I_k in cui cade il campione x (vedi anche §§ 4.3.2 e 10.1.2.4). Prevedere anche per $\hat{x}$ un valore probabilistico generalizza sia il concetto che la trattazione.

[37] Per gli amanti del rigore analitico, il processo logico-matematico che motiva tale definizione viene sviluppato ad esempio al cap. 13 del testo *Elements of Information Theory* di T.M. Cover e J.A. Thomas (1991, Wiley), reperibile ad es. presso
`http://www.cs-114.org/wp-content/uploads/2015/01/Elements_of_Information_Theory_Elements.pdf`

[38] Per $\hat{x}$ incognito, oppure statisticamente indipendente da x.

congiuntamente al vincolo $p(\hat{x}/x) : D_x \leq D$, vincolo conseguibile purché $I(\hat{x};x)$ sia grande a sufficienza.

9.5.2.1 Shannon lower bound

Soluzioni generali per (9.27) sono difficili da ottenere[39], ma si può comunque mostrare che $R(D)$ è continua, monotonicamente decrescente, e convessa (ossia *ad U*). Sussiste inoltre un *limite inferiore* valido per criteri di distorsione $d(x,\hat{x})$ di tipo errore quadratico e per sorgenti qualsiasi e senza memoria, ma che ora illustriamo per il caso continuo, e che afferma

$$R(D) \geq h(X) - h(D) = h(X) - \tfrac{1}{2}\log_2(2\pi \mathrm{e} D) \tag{9.28}$$

dove $h(X)$ è l'entropia differenziale della sorgente, e $h(D)$ quella di una v.a. gaussiana con varianza D. Per dimostrare la (9.28) iniziamo osservando che in base alla relazione $I(X;\hat{X}) = h(X) - h\big(X/\hat{X}\big)$ (§ 9.4.3) possiamo scrivere la (9.27) come

$$\begin{aligned} R(D) &= \min_{D_x \leq D}\big\{I(X;\hat{X})\big\} = \min_{D_x \leq D}\big\{h(X) - h\big(X/\hat{X}\big)\big\} = \\ &= h(X) - \max_{D_x \leq D}\big\{h\big(X/\hat{X}\big)\big\} \end{aligned} \tag{9.29}$$

in cui l'ultimo termine esprime l'incertezza residua su X una volta nota la sua codifica $\hat{X}$, incertezza pari a zero qualora $\hat{X} = X$, e dunque abbiamo $R(D)|_{D=0} = h(X)$. Se invece $D > 0$ possiamo scrivere

$$\max_{D_x \leq D}\big\{h\big(X/\hat{X}\big)\big\} = \max_{D_x \leq D}\big\{h\big(X - \hat{X}/\hat{X}\big)\big\} \leq = \max_{D_x \leq D}\big\{h\big(X - \hat{X}\big)\big\} \tag{9.30}$$

in cui la prima eguaglianza significa che, dopo la conoscenza di $\hat{X}$, rimane la stessa incertezza sia a riguardo del valore vero X che a riguardo dell'errore $X - \hat{X}$[40]; la successiva diseguaglianza deriva invece dalla constatazione che aggiungendo informazione $\big(\hat{X}\big)$ l'entropia non può aumentare, e dunque $h\big(X - \hat{X}/\hat{X}\big) \leq h\big(X - \hat{X}\big)$. Dato però che nella (9.29) l'ultimo termine compare con il segno meno, sostituendovi la (9.30) si ottiene

$$R(D) \geq h(X) - \max_{D_x \leq D}\big\{h\big(X - \hat{X}\big)\big\} \tag{9.31}$$

Per arrivare alla (9.28) è ora sufficiente osservare che una distorsione definita come $D = E\big\{\big(X - \hat{X}\big)^2\big\}$ è a tutti gli effetti una varianza σ^2, dunque la condizione $D_x \leq D$ pone un limite $\sigma^2 \leq D$ alla varianza dell'errore $X - \hat{X}$. Al § 9.3.2 si è mostrato che la sorgente che fornisce la massima entropia differenziale h per σ^2 assegnata è gaussiana, con $h = \frac{1}{2}\log_2\big(2\pi \mathrm{e}\sigma^2\big)$, e dunque $\max_{D_x \leq D}\big\{h\big(X - \hat{X}\big)\big\} \leq \frac{1}{2}\log_2(2\pi \mathrm{e} D)$, che sostituita nella (9.31) fornisce la (9.28).

[39] Benché esista un metodo iterativo di soluzione, vedi

. `https://en.wikipedia.org/wiki/Blahut-Arimoto_algorithm`

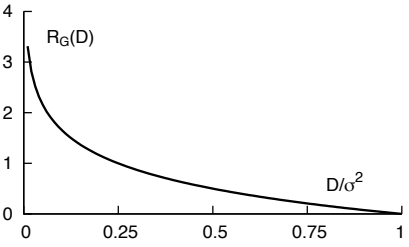
[40] Ciò è conseguenza del fatto che, come osservato a pag. 266, l'entropia differenziale di una v.a. *non dipende* dal suo valore medio, che in questo caso è $\hat{X}$.

9.5.3 Curva velocità-distorsione per sorgente gaussiana

Qualora la v.a. continua x sia di tipo gaussiano, senza memoria e con varianza σ_x^2, inserendo l'espressione della corrispondente entropia differenziale (9.20) nella (9.28) si ottiene una funzione velocità-distorsione (9.27) pari a

$$R_G(D) = \frac{1}{2}\log_2\left(2\pi e\sigma_x^2\right) - \frac{1}{2}\log_2\left(2\pi e D\right) = -\frac{1}{2}\log_2\frac{D}{\sigma_x^2}$$

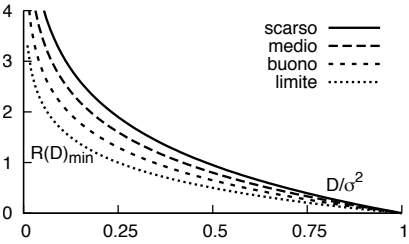
con il segno di uguale[41], ovvero la sorgente gaussiana *consegue* il limite inferiore (9.28). Osserviamo ora che per $D = \sigma_x^2$ si ottiene $R(D)|_{D=\sigma_x^2} = 0$ ovvero *non occorre trasmettere nulla* ($\hat{x} = 0$), in modo che l'errore $e = x - \hat{x} = x$ abbia appunto potenza $\sigma_x^2 = D$. Se poi $D > \sigma_x^2$ ossia la distorsione *è superiore* alla potenza di segnale, è sufficiente generare un valore $\hat{x}$ gaussiano, a media nulla, *indipendente* da x e di varianza $D - \sigma_x^2$ per ottenere una distorsione $E\left\{(x-\hat{x})^2\right\} = \sigma_x^2 + D - \sigma_x^2 = D$.



L'espressione finale per questo caso risulta dunque

$$R_G(D) = \begin{cases} -\frac{1}{2}\log_2\frac{D}{\sigma_x^2} & \text{se} \quad 0 \le D \le \sigma_x^2 \\ 0 & \text{se} \quad D \ge \sigma_x^2 \end{cases} \tag{9.32}$$

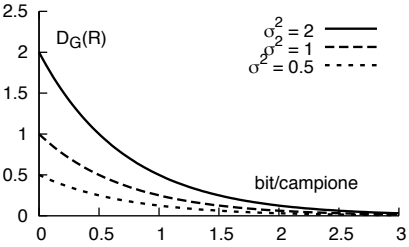
Sorgente non gaussiana e confronto prestazioni In questo caso l'entropia $h(X)$ è inferiore al caso gaussiano, e quindi il limite (9.28) fornisce un valore $R(D)_{min}$ più piccolo di (9.32), *abbassando* la curva mostrata sopra. Una volta stabilito *un modello* (anche sperimentale) della d.d.p. della nostra sorgente, e valutata (anche numericamente) la sua entropia, possiamo usare la (9.28) per tracciare la relativa curva $R(D)_{min}$, e confrontare rispetto ad essa le prestazioni conseguite dal metodo di codifica che stiamo sviluppando, come esemplificato a lato.



Curva distorsione-velocità Invertendo la relazione (9.32) si ottiene[42] che la *minima distorsione* D conseguibile da una sorgente gaussiana in corrispondenza ad una velocità di R binit/campione risulta pari a

$$D_G(R) = 2^{-2R}\sigma_x^2 \tag{9.33}$$

ovvero la distorsione per sorgenti gaussiane e con R fissato è proporzionale alla varianza σ_x^2, e decresce esponenzialmente con la velocità, come mostrato in figura.



[41] Ciò deriva dal considerare $x = \hat{x} + e$ con $\hat{x}$ ed e v.a. gaussiane statisticamente indipendenti, di varianza rispettivamente e $\sigma_x^2 - D$ e D: in tali condizioni si ottiene $E\left\{\left(X-\hat{X}\right)^2\right\} = D$ e $I(X;\hat{X}) = \frac{1}{2}\log_2\frac{\sigma_x^2}{D}$.

[42] $R = -\frac{1}{2}\log_2\frac{D}{\sigma_x^2} \implies -2R = \log_2\frac{D}{\sigma_x^2} \implies 2^{-2R} = \frac{D}{\sigma_x^2} \implies D = 2^{-2R}\sigma_x^2$

Legame con l'SNR Definendo il rapporto segnale-rumore in dB (§ 8.1) come $10 \log_{10} \frac{\sigma_x^2}{D}$ la (9.33) consente di ottenere

$$SNR_{dB} = 10 \log_{10} 2^{2R} = 2R \cdot 10 \log_{10} 2 = 6 \cdot R \quad dB$$

ossia un miglioramento di 6 dB per ogni binit in più utilizzato per la codifica di un campione, confermando in pieno il risultato già ricavato al § 4.3.1.1.

Valori limite Il valore (9.33) rappresenta un *limite superiore* per la distorsione a velocità R per una sorgente non gaussiana, o gaussiana ma con memoria, per la quale si possono ottenere valori di distorsione inferiori; la (9.33) individua quindi *il più grande valore* della distorsione *minima,* per velocità R e potenza σ_x^2 assegnate. D'altra parte è anche definito un *limite inferiore* $D_L(R)$ che individua la minima distorsione sotto cui non si può scendere per un dato R per sorgenti non gaussiane e senza memoria, in modo da poter scrivere

$$D_L(R) = 2^{-2R} Q \quad \leq \quad D(R) \quad \leq \quad D_G(R) = 2^{-2R} \sigma_x^2 \tag{9.34}$$

in cui Q è la...

Potenza entropica Esprime una quantità direttamente legata all'entropia differenziale della sorgente, con valore

$$Q = \frac{1}{2\pi \mathrm{e}} 2^{2h(X)} \tag{9.35}$$

che per sorgenti gaussiane fornisce $Q_G = \sigma_x^2$, mentre per altri tipi di v.a. si ottiene un valore inferiore[43]. Si può anche interpretare Q come la varianza di una sorgente gaussiana con la stessa entropia della sorgente in esame. Sostituendo (9.35) in (9.34) osserviamo come il limite inferiore di distorsione $D_L(R)$ si riduce al diminuire di $h(X)$, ovvero sorgenti meno informative conseguono distorsioni minori a parità di velocità.

9.5.4 Sorgente continua con memoria

Come per il caso di sorgenti discrete anche per quelle continue la dipendenza statistica tra i campioni di segnale riduce la quantità di informazione emessa, cosicché a parità di distorsione la sorgente può essere codificata a velocità ridotta, oppure a parità di velocità si può conseguire una distorsione inferiore. Anche stavolta la sorgente più *difficile* (ossia a cui compete la *massima distorsione minima*) è quella gaussiana, per la quale si ottengono risultati la cui interpretazione è molto interessante; prima di esporli conviene però fare un piccolo passo indietro.

9.5.4.1 Entropia e potenza entropica di sorgente gaussiana con memoria

Analogamente al caso discreto (§ 9.2), per una sequenza (o vettore *colonna)* $\boldsymbol{x} = (x_1, x_2 \cdots, x_n)^T$ di n v.a. con ampiezza continua, descritte da una d.d.p. congiunta $p(\mathbf{x})$, si definisce una entropia differenziale *a blocco* come

$$h_n(X) = -\tfrac{1}{n} \int_{x_1} \int_{x_2} \cdots \int_{x_n} p(\mathbf{x}) \log_2 p(\mathbf{x}) \, d\mathbf{x} \tag{9.36}$$

[43] Ad esempio per una v.a. uniforme si ottiene $Q_U = 0.703 \cdot \sigma_x^2$

la cui valutazione è generalmente intrattabile. Se consideriamo un processo gaussiano $x(t)$ limitato in banda, a media nulla, stazionario e con densità di potenza $\mathcal{P}_x(f)$ *colorata*, i suoi campioni $x_k = x(kT_c)$ (cap. 4) hanno d.d.p. congiunta (§ 6.5)

$$p_X(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^n \det(\boldsymbol{R}_{xx})}} \exp\left\{-\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{R}_{xx}^{-1}\boldsymbol{x}\right\}$$

dove $\boldsymbol{R}_{xx} = E\{\boldsymbol{x}\boldsymbol{x}^\top\}$ è la matrice *di correlazione*, simmetrica e con elementi diagonali uguali tra loro e pari a $\boldsymbol{R}_{xx}(i,i) = E\{x^2\} = \sigma_x^2$. In tal caso la (9.36) fornisce[44]

$$h_n(X) = \frac{1}{2}\log_2\left(2\pi\mathrm{e}\left(\det(\boldsymbol{R}_{xx})\right)^{1/n}\right)$$

da cui, dopo aver definito l'entropia differenziale per simbolo come $h(X) = \lim_{n\to\infty} h_n(X)$, ricaviamo la potenza entropica per questo caso, pari a[45]

$$Q_{G,mem} = \lim_{n\to\infty}\left(\det(\boldsymbol{R}_{xx})\right)^{1/n} = \gamma_x^2\sigma_x^2 \tag{9.37}$$

(vedi § 9.6.3) dove $0 \le \gamma_x^2 \le 1$ è una *misura di piattezza spettrale*[46] che vale uno per un processo *bianco* o senza memoria, tornando così all'espressione $Q = \sigma_x^2$ valida in tal caso. Viceversa Q si riduce ($\gamma_x^2 < 1$) per un processo gaussiano a valori correlati, e quindi con una densità spettrale *colorata* ed una maggiore *predicibilità* dei suoi valori, e dunque una minore entropia.

9.5.4.2 Funzione distorsione-velocità per sorgente gaussiana con memoria

Il valore di Q (9.37) consente il calcolo del *limite inferiore* definito alla (9.34):

$$D_{L,G,mem}(R) = 2^{-2R}Q_{G,mem} = 2^{-2R}\gamma_x^2\sigma_x^2 \tag{9.38}$$

Mostriamo ora in quale caso la distorsione effettiva consegue il limite ossia $D_{G,mem}(R) = D_{L,G,mem}(R)$, e quale sia il valore $D_{G,mem}(R) > D_{L,G,mem}(R)$ in caso contrario.

Il water-filling Le funzioni (9.32) e (9.33) nel caso di sorgente gaussiana con memoria, i cui campioni sono descritti da uno spettro di densità di potenza $S_{xx}(e^{j\omega})$ *colorato*,

[44] La dimostrazione penso sia simile a quella del § 9.3.2, con le complicazioni della notazione matriciale.

[45] Infatti $\lim_{n\to\infty} h_n(X) = \frac{1}{2}\log_2\left(2\pi\mathrm{e}\lim_{n\to\infty}\left(\det(\boldsymbol{R}_x)\right)^{1/n}\right) \quad \Rightarrow$

. $\Rightarrow \quad 2h(X) = \log_2\left(2\pi\mathrm{e}\lim_{n\to\infty}\left(\det(\boldsymbol{R}_x)\right)^{1/n}\right) \quad \Rightarrow \quad 2^{2h(X)} = 2\pi\mathrm{e}\lim_{n\to\infty}\left(\det(\boldsymbol{R}_x)\right)^{1/n} \quad \Rightarrow$

. $\Rightarrow \quad \frac{2^{2h(X)}}{2\pi\mathrm{e}} = Q = \lim_{n\to\infty}\left(\det(\boldsymbol{R}_x)\right)^{1/n}$. Per la seconda uguaglianza della (9.37), si veda il § 9.6.3.

[46] Vedi `https://en.wikipedia.org/wiki/Spectral_flatness`, o meglio N.S. Jayant, P. Noll, *Digital Coding of Waveforms*, 1984 Prentice Hall. Si può mostrare che γ_x^2 può essere interpretato come il rapporto tra la media geometrica e la media aritmetica della densità spettrale di potenza $\mathcal{P}_x(f)$ del processo $x(t)$ limitato in banda $\pm W$: indicando con $S_k = \mathcal{P}_x(f_k)$, $k = 1, 2, \cdots, N$, i campioni equispaziati della densità spettrale valutati a frequenze positive f_k tra zero e la massima frequenza , si ha

$$\gamma_x^2 = \lim_{N\to\infty}\frac{\left(\prod_{k=1}^N S_k\right)^{1/N}}{\frac{1}{N}\sum_{k=1}^N S_k} = \frac{\exp\left(\frac{1}{2W}\int_{-W}^{W}\ln\mathcal{P}_x(f)\,df\right)}{\frac{1}{2W}\int_{-W}^{W}\mathcal{P}_x(f)\,df}$$

Nel caso di un processo bianco, per il quale i valori S_k sono tutti uguali, le due medie coincidono, e $\gamma_x^2 = 1$. Altrimenti, γ_x^2 risulta tanto più piccolo quanto più i valori S_k si discostano dal loro valore medio.

devono essere espresse in forma *parametrica* come[47]

$$D_{G,mem}(\phi) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \min_{\omega}\left[\phi, S_{xx}(e^{j\omega})\right] d\omega \tag{9.39}$$

$$R_{G,mem}(\phi) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \max_{\omega}\left[0, \frac{1}{2}\log_2\frac{S_{xx}(e^{j\omega})}{\phi}\right] d\omega \tag{9.40}$$

in cui all'aumentare di ϕ, D aumenta ed R diminuisce, come andiamo a spiegare con l'aiuto della figura 9.3.

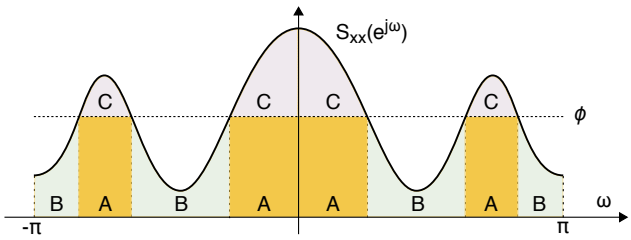


Figura 9.3: Bande di frequenza individuate dal parametro ϕ, ed aree che concorrono ai valori D ed R

Le regioni indicate con B, in cui $\phi > S_{xx}(e^{j\omega})$, non contribuiscono al valore di R, dato che nella (9.40) il log è negativo, mentre contribuiscono al valore di D (eq. (9.39)) solamente per l'area ombreggiata in verde, ossia nelle regioni B la distorsione D ha densità di potenza *colorata* come S_{xx}. Quando invece $S_{xx}(e^{j\omega}) > \phi$ (regioni A) il contributo a D non dipende dal valore di $S_{xx}(e^{j\omega})$ ed assume l'aspetto (aree ocra) di un rumore *bianco*; viceversa (sempre in A) il contributo ad R è legato (con legge logaritmica) al valore delle aree lilla (indicate con C).

Questa interpretazione grafica delle (9.39) e (9.40) viene denominata *a riempimento d'acqua* perché simula un recipiente con la forma di $S_{xx}(e^{j\omega})$ per il quale le regioni A fungono da *vasi comunicanti* riempiti di acqua fino al livello ϕ. Da tale discussione traiamo i risultati

- una volta assegnata una distorsione D la codifica ottima *non spreca* velocità binaria R per rappresentare regioni di frequenza (B) dove il segnale è più debole;
- la costanza delle densità spettrale dell'errore D nelle regioni A fa si che l'SNR *locale* migliori proporzionalmente a $S_{xx}(e^{j\omega})$, densità spettrale del segnale.

Osserviamo ora che ponendo $\phi > \max\left\{S_{xx}(e^{j\omega})\right\}$ la sagoma di $S_{xx}(e^{j\omega})$ *si riempie completamente d'acqua*, la (9.39) fornisce $D = \sigma_x^2$, mentre la (9.40) restituisce $R = 0$; viceversa nel caso in cui $\phi < \min\left\{S_{xx}(e^{j\omega})\right\}$ ci si trova nelle condizioni di *bassa distorsione* per le quali D consegue il suo valore limite inferiore espresso dalla (9.38), ovvero

$$D_{G,mem,low-dist}(R) = 2^{-2R}\gamma_x^2\sigma_x^2 \tag{9.41}$$

dove $0 \le \gamma_x^2 \le 1$ è la misura di *piattezza spettrale* già introdotta nella (9.37). Qualora $S_{xx}(e^{j\omega})$ sia costante risulta $\gamma_x^2 = 1$, ri-ottenendo così il risultato (9.33) trovato per il caso senza memoria.

Esempio Consideriamo la sequenza di tipo *autoregressivo* $x(n) = z(n) + \alpha \cdot x(n-1)$ presente all'uscita di un filtro passa basso numerico IIR del primo ordine (§ 5.3.2.2) con $h(n) = \alpha^n$

[47] Di questo non viene fornita dimostrazione, di cui trovo indicazione essere presente in T. Berger, *Rate distortion theory*, Prentice-Hall 1971, che non trovo pubblicamente disponibile in rete.

(per $n \geq 0$ ed $0 < \alpha < 1$), al cui ingresso è presente una sequenza $z(n)$ di campioni di un processo gaussiano, a media nulla, varianza σ_z^2 ed a valori indipendenti ed incorrelati da quelli di x. Anche $x(n)$ sarà dunque gaussiano, ma con autocorrelazione[48] $\mathcal{R}_{xx}(k) = \alpha^{|k|}\sigma_x^2$, varianza $\sigma_x^2 = \frac{\sigma_z^2}{1-\alpha^2}$, piattezza spettrale $\gamma_x^2 = 1 - \alpha^2$ e densità di potenza *colorata* pari a $S_{xx}\left(e^{j\omega}\right) = \sigma_z^2 \cdot |H_{xx}\left(e^{j\omega}\right)|^2 = \sigma_x^2 \frac{1-\alpha^2}{1+\alpha^2-2\alpha\cos\omega}$. All'aumentare di α la misura di piattezza γ_x^2 si riduce, $x(n)$ diviene più predicibile, e la distorsione $D = 2^{-2R}(1-\alpha^2)\sigma_x^2$ (9.38) diminuisce.

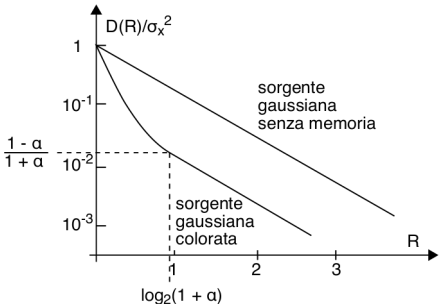
Il minimo valore di $S_{xx}\left(e^{j\omega}\right)$ si ottiene per $\omega = \pi$ (vedi fig. 4.13), ed è pari a

$$\min_{\omega}\left\{S_{xx}(e^{j\omega})\right\} = \sigma_x^2 \frac{1-\alpha^2}{1+\alpha^2-2\alpha\cos\omega}\bigg|_{\omega=\pi} = \sigma_x^2 \frac{1-\alpha^2}{(1+\alpha)^2} = \sigma_x^2 \frac{1-\alpha}{1+\alpha} \tag{9.42}$$

Ponendo (ad esempio) $\alpha = 0.95$, la (9.42) fornisce $0.0256 \cdot \sigma_x^2$: pertanto la regione *a bassa distorsione* in questo caso è definita come $D/\sigma_x^2 \leq 0.0256$, o $R \geq \log_2(1+\alpha) = 0.964$[49], ed in tale regione è lecito applicare la (9.41) ossia

$$D(R) = 2^{-2R}(1-\alpha^2)\sigma_x^2 = 0.0975 \cdot 2^{-2R}\sigma_x^2$$

avendovi sostituito i valori per γ_x^2 ed α. Il risultato è la retta parallela a quella per una sorgente gaussiana senza memoria mostrata alla figura a lato, su di una scala logaritmica per le distorsioni. Per valori $D/\sigma_x^2 > 0.0256$ la distorsione D aumenta più rapidamente ed il suo valore va determinato applicando la (9.39), per raggiungere (ad $R = 0$) il valore σ_x^2, come avviene per il caso senza memoria.



9.5.4.3 Sorgente non gaussiana

In questo caso la (9.41) si riscrive sostituendo al posto di σ_x^2 la potenza entropica Q espressa dalla (9.35) in cui il valore di entropia differenziale è ora quello della sorgente con memoria, inferiore al caso senza memoria, ottenendo così valori $D(R)$ ancora inferiori.

L'applicazione dei principi relativi alla codifica di sorgente al caso specifico dei messaggi multimediali (audio e video) viene trattata al capitolo 10, mentre l'applicazione dei concetti di informazione mutua media e di entropia condizionale al calcolo della capacità di canale è sviluppata al capitolo 17.

9.6 Appendici

[48] I risultati indicati sono derivati al § 9.6.4

[49] Infatti la (9.42) può essere riarrangiata come $D/\sigma_x^2 = \frac{1-\alpha}{1+\alpha}$ mentre dalla (9.41) si ottiene $D/\sigma_x^2 = 2^{-2R}(1-\alpha^2)$; dunque $\frac{1-\alpha}{1+\alpha} = 2^{-2R}(1-\alpha^2) \Rightarrow 2^{-2R} = \frac{1-\alpha}{1+\alpha}\frac{1}{1-\alpha^2} = \frac{1}{(1+\alpha)^2}$ e quindi infine $R = -\frac{1}{2}\log_2\frac{1}{(1+\alpha)^2} = \log_2\sqrt{(1+\alpha)^2} = \log_2(1+\alpha)$

9.6.1 Metodo dei moltiplicatori di Lagrange

Costituisce un modo di affrontare un *problema* di ottimizzazione *vincolata*[50], ossia individuare i punti $\boldsymbol{x}^*$ di massimo o minimo (detti *estremali*) per una funzione *obiettivo* $f(\boldsymbol{x})$ definita in un aperto $\mathcal{A} \subset \mathbb{R}^n$ (ossia dipendente da più variabili $\boldsymbol{x} = (x_1, x_2, \cdots x_n)$), nel rispetto di una o più condizioni di *vincolo* del tipo $q_i(\boldsymbol{x}) = b_i$ ossia $g_i(\boldsymbol{x}) = 0$ con $i = 1, 2, \cdots, m$ dopo aver posto $g_i(\boldsymbol{x}) = q_i(\boldsymbol{x}) - b_i$.

Le condizioni di vincolo individuano una *regione ammissibile* $\mathcal{X}$ per le soluzioni $\boldsymbol{x}^* \in \mathcal{X}$ come $\mathcal{X} = \{x \in \mathcal{A} : g_i(\boldsymbol{x}) = 0 \text{ con } i = 1, 2, \cdots, m\}$.

Osservazione L'insieme $\mathcal{X}$ ha gradi di libertà ridotti rispetto all'aperto $\mathcal{A}$: ad esempio se $n = 2$, ovvero $z = f(x, y)$ è una superficie, allora un vincolo $g(x, y) = 0$ è una *curva* nel piano (x, y), proiezione della *intersezione* tra la superficie $z = q(x, y)$ ed il piano orizzontale $z = b$. A sua volta la curva espressa dal vincolo $g(x, y) = 0$ viene *proiettata* in verticale sulla superficie $z = f(x, y)$, individuando su di essa *un cammino* lungo il quale si trovano i punti $\boldsymbol{x}^*$ di ottimo vincolato. Qualora si aggiunga un ulteriore vincolo $g_2(x, y) = 0$ l'insieme $\mathcal{X}$ si riduce ulteriormente ai punti di intersezione tra i cammini disegnati sulla superficie $z = f(x, y)$ dai due vincoli.

A patto che le funzioni siano derivabili con derivate continue e che $m < n$, trovare i punti $\boldsymbol{x}^*$ estremali di $f(\boldsymbol{x})$ nel rispetto dei vincoli $\boldsymbol{g}(\boldsymbol{x}) = 0$ passa per la definizione della funzione *Lagrangiana* come

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) - \left\langle \boldsymbol{\lambda}, \boldsymbol{g}(\boldsymbol{x}) \right\rangle = f(\boldsymbol{x}) - \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}) \tag{9.43}$$

in cui $\boldsymbol{g}(\boldsymbol{x})$ è il vettore delle m funzioni $g_i(\boldsymbol{x})$, $\boldsymbol{\lambda}$ un vettore di m *moltiplicatori* scalari, e $\left\langle \boldsymbol{\lambda}, \boldsymbol{g}(\boldsymbol{x}) \right\rangle$ esprime il prodotto scalare (§ 2.4.3) tra i due. Osserviamo che $\mathcal{L}$ è una funzione di $n + m$ variabili, le dimensioni di $\boldsymbol{x}$ e $\boldsymbol{\lambda}$.

Gradiente prima di poter proseguire occorre definire l'operatore *gradiente* ∇ di $f(\boldsymbol{x})$ come il vettore i cui elementi ne sono le derivate parziali, ossia $\nabla f(\boldsymbol{x}) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \cdots, \frac{\partial f}{\partial x_n} \right)$. Il suo orientamento nello spazio indica la direzione verso cui $f(\boldsymbol{x})$ cresce più velocemente, e se $\boldsymbol{x} = \boldsymbol{x}_0$ è un *punto stazionario* (minimo, massimo o di sella) di $f(\boldsymbol{x})$, allora $\nabla f(\boldsymbol{x}_0) = \boldsymbol{0}$, ovvero piccole variazioni dell'argomento $\boldsymbol{x} \simeq \boldsymbol{x}_0$ non alterano di molto il valore di f.

Possiamo finalmente enunciare il teorema, che afferma

> *Qualora esista un punto $\boldsymbol{x}^*$ di ottimo vincolato, e la matrice Jacobiana $\mathbf{J}\{\boldsymbol{g}(\boldsymbol{x}^*)\}$ abbia rango m*[51]*, allora esiste un vettore $\boldsymbol{\lambda}^* \in \mathbb{R}^m$ tale che*
>
> $$\nabla \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) = \boldsymbol{0} \tag{9.44}$$
>
> *ovvero i valori $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ individuano punti stazionari di* (9.43)

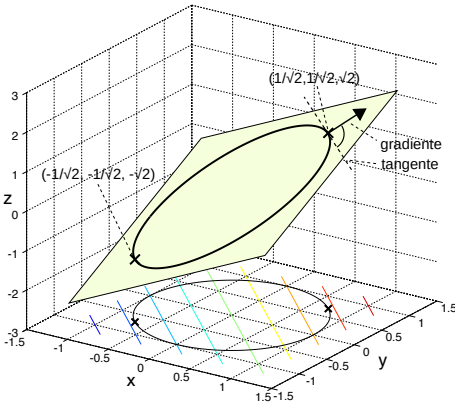
Ciò comporta che

[50] Vedi ad es. `https://en.wikipedia.org/wiki/Lagrange_multiplier`

[51] La matrice Jacobiana $\mathbf{J}\{\boldsymbol{g}(\boldsymbol{x})\}$ (associata ai vincoli $g_i(\boldsymbol{x})$) è stata introdotta al 6.4.2, ed essendo la sua i–esima riga riga ottenuta come la sequenza delle derivate parziali $\frac{\partial g_i}{\partial x_j}$, si ottiene *impilando* i vettori gradiente $\nabla_x g_i(\boldsymbol{x})$ calcolati per ciascun vincolo.

1. l'azzeramento di $\nabla\mathcal{L}(\boldsymbol{x},\boldsymbol{\lambda})$ per un qualche valore $(\boldsymbol{x}^*,\boldsymbol{\lambda}^*)$ è condizione *necessaria* all'esistenza del punto $\boldsymbol{x}^*$ di ottimo, ovvero i valori che verificano la (9.44) possono *non essere* soluzione al problema di ottimo, ma devono essere verificati singolarmente;
2. $\nabla\mathcal{L}$ (9.44) è un vettore i cui elementi sono $n+m$ funzioni di $\boldsymbol{x}$ e $\boldsymbol{\lambda}$, il cui azzeramento realizza un sistema di altrettante equazioni, le cui soluzioni sono i valori $(\boldsymbol{x}^*,\boldsymbol{\lambda}^*)$ desiderati;
3. i primi n elementi di $\nabla\mathcal{L}$ sono le $\frac{\partial\mathcal{L}}{\partial x_j}$ con $j=1,2,\cdots,n$, il cui azzeramento implica $\frac{\partial f}{\partial x_j}=\sum_{i=1}^{m}\lambda_i^*\frac{\partial g_i}{\partial x_j}$ ovvero $\nabla_{\boldsymbol{x}}f(\boldsymbol{x}^*)=\sum_{i=1}^{m}\lambda_i^*\nabla_{\boldsymbol{x}}g_i(\boldsymbol{x}^*)$ od anche $\nabla_x f(\boldsymbol{x}^*)=\left\langle\boldsymbol{\lambda}^*,\nabla_x\boldsymbol{g}(\boldsymbol{x}^*)\right\rangle$, e cioè quando $(\boldsymbol{x},\boldsymbol{\lambda})=(\boldsymbol{x}^*,\boldsymbol{\lambda}^*)$ il gradiente della funzione obiettivo $\nabla_{\boldsymbol{x}}f$ diviene eguale ad una combinazione lineare $\sum_{i=1}^{m}\lambda_i^*\nabla_{\boldsymbol{x}}g_i$ degli m gradienti delle condizioni di vincolo, con i moltiplicatori λ_j^* come coefficienti. Ciò comporta che $\nabla_{\boldsymbol{x}}f(\boldsymbol{x}^*)$ *è ortogonale* a ciascun cammino individuato dai vincoli[52], o ovvero che la curva di livello di $f(\boldsymbol{x})$ che passa per $\boldsymbol{x}=\boldsymbol{x}^*$ è *tangente* (in tale punto) alla curva descritta dal vincolo;
4. gli ultimi m elementi di $\nabla\mathcal{L}$ corrispondono a $\frac{\partial\mathcal{L}}{\partial\lambda_j}=-g_j(\boldsymbol{x})$ e dunque il loro annullamento comporta il rispetto dei vincoli;
5. se i vincoli sono soddisfatti il termine $\sum_{i=1}^{m}\lambda_i^*g_i(\boldsymbol{x})$ della (9.43) è nullo, e dunque i punti estremali di $\mathcal{L}$ al variare di $\boldsymbol{x}$ lo sono anche per la $f(\boldsymbol{x})$ di partenza;
6. la condizione $J\{\boldsymbol{g}(\boldsymbol{x}^*)\}$ *di rango m* garantisce che i gradienti dei vincoli siano linearmente indipendenti, consentendo di trarre le conclusioni di cui al punto 3. Qualora la condizione non si verifichi la soluzione offerta dal teorema non è praticabile, dato che in tal caso $\nabla\mathcal{L}$ può non azzerarsi nei punti di ottimo, per qualunque $\boldsymbol{\lambda}$.

Esempio Si trovino i punti di massimo e di minimo della funzione $f(x,y)=x+y$ sottoposta al vincolo $g(x,y)=x^2+y^2-1=0$. Con l'aiuto della figura, osserviamo che i punti devono giacere sulla intersezione tra un piano inclinato $z=x+y$ e la proiezione su di esso di una circonferenza di raggio unitario, risultato dell'intersezione tra un paraboloide $z=x^2+y^2$ ed il piano $z=1$. Per la Lagrangiana otteniamo $\mathcal{L}(x,y,\lambda)=x+y-\lambda\left(x^2+y^2-1\right)$, e l'annullamento del relativo gradiente



da luogo al sistema $\begin{cases}2\lambda x-1=0\\2\lambda y-1=0\\x^2+y^2-1=0\end{cases}$, da cui si può ottenere che $(x^*,y^*,\lambda^*)=\pm\left(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$. Dato che $f\left(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)=\sqrt{2}$ e che $f\left(-\frac{1}{\sqrt{2}},-\frac{1}{\sqrt{2}}\right)=-\sqrt{2}$, il primo punto

[52] Vedi ad es. `https://www.geogebra.org/m/gXyun8mD`

individua un massimo, ed il secondo un minimo, entrambi di tipo vincolato. Osserviamo come ∇f sia ovunque pari a $(1,1)$, e nei punti di ottimo risulti *ortogonale* alla tangente al cammino individuato dal vincolo.

9.6.2 Massimo dell'entropia per variabile aleatoria gaussiana

Si intende ora mostrare la validità della (9.21), ovvero che l'entropia differenziale per sorgente continua gaussiana $h_G = \frac{1}{2}\log_2\left(2\pi e\sigma^2\right)$ (eq. (9.20)) è il *massimo* che si può ottenere per qualunque scelta della d.d.p. di primo ordine $p(x)$ della sorgente, una volta fissata la sua varianza σ^2, ossia

$$h_G = \frac{1}{2}\log_2\left(2\pi e\sigma^2\right) = \max_{p(x)}\left\{-\int p(x)\log_2 p(x)\,dx\right\} \quad \text{dato} \quad \sigma^2$$

Per arrivare allo scopo si può adottare[53] il metodo dei moltiplicatori di Lagrange (§ 9.6.1), considerando $p(x) = p$ come una *variabile* p (anziché una funzione) rispetto a cui massimizzare, nel rispetto dei vincoli $g_1(p) = \int p(x)\,dx - 1 = 0$ e $g_2(p) = \int x^2 p(x)\,dx - \sigma^2 = 0$, mentre il vincolo sul valor medio è ininfluente a causa dell'invarianza evidenziata a pag. 266.

Scriviamo dunque il lagrangiano come

$$\begin{aligned}\mathcal{L}(p,\lambda_1,\lambda_2) &= -\int p(x)\log_2 p(x)\,dx - \lambda_1\left(\int p(x)\,dx - 1\right) - \lambda_2\left(\int x^2 p(x)\,dx - \sigma_x^2\right) = \\ &= -\int\left[p(x)\log_2 p(x) + \lambda_1 p(x) + \lambda_2 x^2 p(x)\right]dx + \lambda_1 + \lambda_2\sigma_x^2 = \\ &= -\tfrac{1}{\ln 2}\int\left[p(x)\ln p(x) + \lambda_1 p(x) + \lambda_2 x^2 p(x)\right]dx + \lambda_1 + \lambda_2\sigma_x^2\end{aligned}$$

in cui si è sostituito $\log_2 p(x) = \frac{1}{\ln 2}\ln p(x)$ e si adotta una eguale scalatura per i moltiplicatori λ_i. Il massimo di $\mathcal{L}(p,\lambda_1,\lambda_2)$ si ottiene eguagliandone a zero le derivate parziali, ed applicando la proprietà di *derivata sotto il segno di integrale*[54] scriviamo

$$\frac{\partial\mathcal{L}}{\partial p} = -\tfrac{1}{\ln 2}\int\left[\ln p(x) - p(x)\,\frac{1}{p(x)} + \lambda_1 + \lambda_2 x^2\right]dx = 0$$

in cui si è applicata la regola della derivata del prodotto $p\ln p$. L'uguaglianza con zero si verifica se $\ln p(x) - 1 + \lambda_1 + \lambda_2 x^2 = 0$ ovvero $\ln p(x) = 1 - \lambda_1 - \lambda_2 x^2$, da cui

$$p(x) = \mathrm{e}^{1-\lambda_1-\lambda_2 x^2} = \mathrm{e}^{1-\lambda_1}\mathrm{e}^{-\lambda_2 x^2} = \mathrm{e}^{\alpha}\mathrm{e}^{-\beta x^2}$$

dove $\alpha = 1-\lambda_1$ e $\beta = \lambda_2$ deve essere positivo per poter avere $\int p(x)\,dx$ finito.

Osserviamo ora che il vincolo $\int p(x)\,dx = 1$ determina che[55] $\int \mathrm{e}^{\alpha}\mathrm{e}^{-\beta x^2}dx = \mathrm{e}^{\alpha}\sqrt{\pi/\beta} = 1$ e dunque la condizione $g_1(p) = 0$ è soddisfatta qualora

$$e^{\alpha} = \sqrt{\beta/\pi}$$

[53] Un metodo alternativo è mostrato in `https://en.wikipedia.org/wiki/Differential_entropy`, mentre `https://kconrad.math.uconn.edu/blurbs/analysis/entropypost.pdf` approfondisce la questione.

[54] Considerando la derivata come limite di un rapporto incrementale, si tratta di una conseguenza di `https://it.wikipedia.org/wiki/Passaggio_al_limite_sotto_segno_di_integrale`

[55] Infatti sappiamo che $\int \frac{1}{\sqrt{2\pi}\sigma_x}\mathrm{e}^{-\frac{x^2}{2\sigma_x^2}}dx = 1$ ovvero, ponendo $1/2\sigma_x^2 = \beta$ e dunque $\sigma_x^2 = 1/2\beta$ otteniamo $\int \mathrm{e}^{-\beta x^2}dx = \sqrt{2\pi\sigma_x^2} = \sqrt{2\pi/2\beta} = \sqrt{\pi/\beta}$.

Infine, il vincolo $\int x^2 p(x)\,dx = \sigma^2$ impone che

$$\int x^2 \mathrm{e}^{\alpha}\mathrm{e}^{-\beta x^2} dx = \sqrt{\beta/\pi}\int x^2 \mathrm{e}^{-\beta x^2} dx = \sigma^2$$

che risulta soddisfatta qualora $\beta = 1/2\sigma^2$, ottenendo infatti in tal modo per $p(x)$ l'espressione di una gaussiana ovvero

$$p(x) = \mathrm{e}^{\alpha}\mathrm{e}^{-\beta x^2} = \sqrt{\beta/\pi}\mathrm{e}^{-\beta x^2}\Big|_{\beta=1/2\sigma^2} = \frac{1}{\sqrt{2\pi\sigma^2}}\mathrm{e}^{-\frac{x^2}{2\sigma^2}}$$

Notiamo esplicitamente la differenza rispetto al caso continuo, in cui la d.d.p. che rende massima l'entropia è invece quella uniforme. Presso Wikipedia è possibile trovare un elenco[56] di valori di entropia differenziale calcolata per diverse scelte di d.d.p.

9.6.3 Misura di piattezza spettrale di processo gaussiano

Per dimostrare[57] la seconda eguaglianza della (9.37) occorre prima mettere in evidenza alcune proprietà degli *autovalori* $\lambda_i, i = 1, 2\cdots, n$ della matrice di correlazione $\boldsymbol{R}_{xx}$ simmetrica, definita positiva, di dimensioni $n \times n$, e con valori $\boldsymbol{R}_{xx}(i,j) = E\{x_i x_j\}$ valutati per i campioni x_i di un processo a media nulla, e dunque pari a σ_x^2 sulla diagonale.

Come noto da altri corsi, ad ogni autovalore è associato il relativo *autovettore* $\boldsymbol{v}_i$ dalla relazione $\boldsymbol{R}_{xx}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$, e l'insieme degli autovalori può essere trovato risolvendo *l'equazione caratteristica* $\det(\boldsymbol{R}_{xx} - \lambda \boldsymbol{I}) = \boldsymbol{0}$ dove $\boldsymbol{I}$ è la matrice *identità*, diagonale $n \times n$. Tale equazione è un polinomio di grado n in λ e le peculiarità di $\boldsymbol{R}_{xx}$ assicurano di poter trovare n zeri reali, a partire dai quali possono essere trovati gli autovettori corrispondenti a meno di un fattore di scala, che conviene fissare in modo che gli autovettori siano *ortonormali*, ossia per essi valga la relazione $\boldsymbol{v}_i^{\top}\boldsymbol{v}_j = \delta_{ij}$. Sussistono ora le relazioni

1. $\boldsymbol{R}_{xx} = \sum_{i=1}^{n} \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^{\top}$
2. $\det(\boldsymbol{R}_{xx}) = \prod_{i=1}^{n} \lambda_i$
3. $\mathrm{tr}(\boldsymbol{R}_{xx}) = \sum_{i=1}^{n} \lambda_i$ ma dato che è anche vero che $\mathrm{tr}(\boldsymbol{R}_{xx}) = n\sigma_x^2$, per qualunque n il valor medio degli autovalori eguaglia la varianza, ovvero
4. $\sigma_x^2 = \frac{1}{n}\sum_{i=1}^{n} \lambda_i$. Essendo quindi gli autovalori una misura di varianza, sono non-negativi. Inoltre, se λ_i è autovalore di $\boldsymbol{R}_{xx}$, allora $1/\lambda_i$ è autovalore di $\boldsymbol{R}_{xx}^{-1}$.

A questo punto il *colpo di scena* deriva dall'applicazione del *teorema di distribuzione di Toeplitz (teorema di Szego[58])* che afferma che per qualunque funzione $f(.)$

$$\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} f(\lambda_i) = \frac{1}{2\pi}\int_{-\pi}^{\pi} f\left[S_{xx}(e^{j\omega})\right] d\omega \tag{9.45}$$

[56] Vedi `https://en.wikipedia.org/wiki/Differential_entropy`

[57] La trattazione segue quella fornita in N.S. Jayant, P. Noll, *Digital Coding of Waveforms*, 1984 Prentice Hall.

[58] In base a quanto riportato presso `https://en.wikipedia.org/wiki/Szego_limit_theorems` il teorema si applica quando gli autovalori sono quelli di una matrice di Toeplitz, i cui elementi sono coefficienti di Fourier di una funzione definita sul cerchio unitario, esattamente come nel nostro caso.

in cui $S_{xx}(e^{j\omega}) = \sum_{k=-\infty}^{\infty} \boldsymbol{R}_{xx}(k)\, e^{-jk\omega}$ è la DTFT (§ 4.4) della funzione di autocorrelazione $\boldsymbol{R}_{xx}(k) = E\{x_n x_{n+k}\}$ cui valori compaiono nella matrice $\boldsymbol{R}_{xx}$„ calcolata per la sequenza $\{x_n\}$, di cui $S_{xx}(e^{j\omega})$ è lo spettro di densità di potenza. In base all'osservazione 4., qualora $f(.)$ sia pari ad una identità la (9.45) diviene un aspetto del teorema di Parseval, e calcola la potenza di $\{x_n\}$. Ma ora si compie *una magia*, poiché ponendo invece $f(.) = \ln(.)$ l'eq. (9.45) diventa

$$\lim_{n\to\infty} \ln\left[\prod_{i=1}^{n} \lambda_i\right]^{1/n} = \frac{1}{2\pi}\int_{-\pi}^{\pi} \ln\left[S_{xx}(e^{j\omega})\right] d\omega \tag{9.46}$$

da cui in base all'osservazione 2 ne consegue[59]

$$\lim_{n\to\infty} \left(\det(\boldsymbol{R}_{xx})\right)^{1/n} = \exp\left\{\frac{1}{2\pi}\int_{-\pi}^{\pi} \ln\left[S_{xx}(e^{j\omega})\right] d\omega\right\} = \eta_x^2$$

dove η_x^2 individua la varianza del *minimo errore di predizione*, ossia la varianza del processo bianco in ingresso ad un filtro autoregressivo (vedi § 10.1.2.2) alla cui uscita si ottiene un segnale con autocorrelazione $\boldsymbol{R}_{xx}(k)$. Dividendo η_x^2 per $\sigma_x^2 = E\left\{x^2\right\} = \frac{1}{2\pi}\int_{-\pi}^{\pi} S_{xx}(e^{j\omega}) d\omega$ otteniamo la *misura di piattezza spettrale*

$$\gamma_x^2 = \frac{\eta_x^2}{\sigma_x^2} = \frac{\exp\left\{\frac{1}{2\pi}\int_{-\pi}^{\pi} \ln\left[S_{xx}(e^{j\omega})\right] d\omega\right\}}{\frac{1}{2\pi}\int_{-\pi}^{\pi} S_{xx}(e^{j\omega}) d\omega} \tag{9.47}$$

che compare nella (9.37).

9.6.4 Autocorrelazione di sequenza autoregressiva

Sviluppiamo qui i passaggi necessari ad ottenere le grandezze indicate nell'esempio di pag. 277, osservando innanzitutto che il processo $x(n) = z(n) + \alpha \cdot x(n-1)$ viene anche detto *Markoviano* di primo ordine, dato che la dipendenza statistica dal passato si estende al solo campione precedente. Come osservato, $x(n)$ è l'uscita di un filtro IIR del primo ordine al cui ingresso sono posti campioni di un processo gaussiano bianco a media nulla con varianza σ_z^2.

Per il valore di autocorrelazione, come noto (§ 7.4) risulta

$$\mathcal{R}_{xx}(k) = \mathcal{R}_{hh}(k) * \mathcal{R}_{zz}(k) = \sigma_z^2 \mathcal{R}_{hh}(k)$$

in quanto $\mathcal{R}_{zz}(k) = \sigma_z^2 \delta(k)$. Con riferimento alla figura che segue, per $k \geq 0$ si ha

$$\begin{aligned}\mathcal{R}_{hh}(k) &= \sum_{n=0}^{\infty} h(n)\, h(n+k) = \sum_{n=0}^{\infty} \alpha^n \alpha^{n+k} = \\ &= \alpha^k \sum_{n=0}^{\infty} \alpha^{2n} = \frac{\alpha^k}{1-\alpha^2}\end{aligned}$$

mentre per $k < 0$

$$\begin{aligned}\mathcal{R}_{hh}(k) &= \sum_{n=|k|}^{\infty} h(n)\, h(n+k) = \sum_{n=|k|}^{\infty} \alpha^n \alpha^{n+k} = \\ &= \alpha^{-k} \sum_{n=|k|}^{\infty} \alpha^{2(n+k)} = \alpha^{-k} \sum_{n=0}^{\infty} \alpha^{2n} = \frac{\alpha^{-k}}{1-\alpha^2}\end{aligned}$$

[59] Infatti indicando il secondo membro di (9.46) con α otteniamo

$\lim_{n\to\infty} \ln\left[\prod_{i=1}^{n} \lambda_i\right]^{1/n} = \lim_{n\to\infty} \ln\left(\det(\boldsymbol{R}_{xx})^{1/n}\right) = \alpha$ e dunque $e^{\alpha} = \lim_{n\to\infty} \det(\boldsymbol{R}_{xx})^{1/n}$

ma dato che se $k < 0$ allora $-k = |k|$, possiamo scrivere

$$\mathcal{R}_{hh}(k) = \frac{\alpha^{|k|}}{1-\alpha^2}$$

per $\forall k$. Risulta quindi

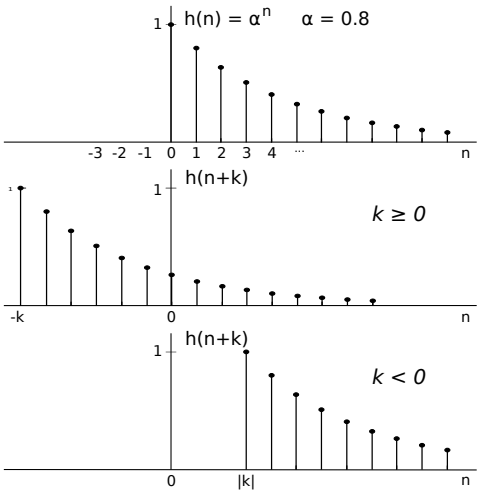
$$\mathcal{R}_{xx}(k) = \sigma_z^2 \mathcal{R}_{hh}(k) = \sigma_z^2 \frac{\alpha^{|k|}}{1-\alpha^2}$$

a cui corrisponde una varianza

$$\sigma_x^2 = \mathcal{R}_{xx}(0) = \sigma_z^2 \frac{1}{1-\alpha^2}$$

ottenendo in definitiva

$$\mathcal{R}_{xx}(k) = \sigma_z^2 \alpha^{|k|}$$



Applicando poi la definizione (9.47) di piattezza spettrale otteniamo

$$\gamma_x^2 = \frac{\sigma_z^2}{\sigma_x^2} = \sigma_z^2 \frac{1-\alpha^2}{\sigma_z^2} = 1-\alpha^2$$

mentre la risposta in frequenza $H_{xx}\left(e^{j\omega}\right)$ è stata già ricavata a pag. **??**, eq. (**??**).

L'opera

Trasmissione dei Segnali e Sistemi di Telecomunicazione

è il risultato di un progetto ventennale di cultura libera, aggiornato di continuo ed evolutosi fino alla forma attuale. La sua disponibilità pubblica è regolata dalle norme di licenza Creative Commons

Attribuzione - Non commerciale - Condividi allo stesso modo

`https://creativecommons.org/licenses/by-nc-sa/4.0/deed.it`

e tutte le risorse relative al testo sono accessibili presso

`https://teoriadeisegnali.it/libro/`

Puoi contribuire al suo successo promuovendone la diffusione e supportarne lo sviluppo attraverso una donazione, in buona parte devoluta ai progetti *open source*[1] che ne hanno resa possibile realizzazione e divulgazione. Ai donatori viene accordato un accesso *vitalizio* al formato PDF *navigabile* di tutte le edizioni presenti *e future*.

[1]
- Lyx - `http://www.lyx.org/`
- LaTeX- `https://www.latex-project.org/`
- TeX Users Group - `https://tug.org/`
- Inkscape - `http://www.inkscape.org/`
- Gnuplot - `http://www.gnuplot.info/`
- Octave - `http://www.gnu.org/software/octave/`
- Geany - `https://www.geany.org/`
- Linux - `https://www.linux.it/`
- Free Software Foundation - `https://shop.fsf.org/`
- GNOME Foundation - `https://www.gnome.org/`
- Mozilla Foundation - `https://www.mozilla.org/it/`
- Wikipedia - `https://it.wikipedia.org`
- Internet Archive - `https://archive.org/about/`
- Creative Commons - `https://creativecommons.it/chapterIT/`
- WordPress - `https://it.wordpress.org/`
- Phplist - `https://www.phplist.org/`