Appunti di Elaborazione di Segnali Multimediali
a.a. 2013/2014

Compressione di immagini

G.Poggi, L.Verdoliva

Questi appunti hanno come obiettivo quello di fornire i concetti base sulla compressione, con particolare attenzione alle tecniche basate su trasformata. Innanzitutto, si introduce il problema della *codifica di sorgente*, spesso impropriamente detta compressione dati, che ha come obiettivo quello di convertire un segnale numerico in una rappresentazione più compatta, in modo da minimizzare le risorse richieste per la sua trasmissione o immagazzinamento. Questa nuova rappresentazione può conservare rigorosamente il contenuto informativo dei dati (codifica *lossless*) oppure, allo scopo di aumentare il livello di compressione, consentire un certo livello di distorsione (codifica *lossy*). La scelta fra queste due opzioni avviene in base alla natura dei dati da codificare (es. analogica o numerica), ed al grado di compressione e di fedeltà richiesto dall'applicazione.

Quindi vengono descritti i metodi maggiormente usati per lo sviluppo di un efficiente algoritmo di codifica lossy, basati principalmente sulla quantizzazione, e di fatto su una pre-elaborazione dei dati, che di solito è una trasformazione. Questa operazione ha l'interessante proprietà di decorrelare i dati e di compattare l'energia in pochi coefficienti, rendendo così più efficiente il processo di quantizzazione. La trasformata ottima dal punto di vista della compressione è quella di Karhunen-Loéve (KLT), tuttavia a causa della sua complessità nella pratica quella che viene effettivamente usata per esempio nello standard JPEG è la trasformata coseno discreta (DCT).

1 La codifica di sorgente

Se consideriamo un'immagine a colori di dimensioni 512×512 pixel, le risorse necessarie per la sua memorizzazione, nell'ipotesi in cui ogni pixel sia rappresentato con 24 bit, sono date da $512 \times 512 \times 24 = 6291456$ bit. Sebbene la tecnologia abbia fatto notevoli passi avanti nell' immagazzinare volumi sempre maggiori di informazione (es. CD-ROM di alta capacità), la richiesta di memoria aumenta in modo anche più veloce. I dati devono poi essere distribuiti agli utenti finali spesso attraverso le ordinarie reti di telecomunicazione. Le connessioni internet trasferiscono dati a una velocità che va da 56 kb/s (per le linee telefoniche tradizionali) fino a più di 13 Mb/s (per linee a larga banda), con un tempo di trasmissione per l'immagine considerata che varia quindi da quasi 2 minuti a 0.5 secondi circa. La compressione può ridurre il tempo di trasmissione (e quindi i costi relativi) anche più di 10 volte, e anche se aumentasse la banda disponibile, le risorse necessarie risulterebbero sempre insufficienti rispetto alla domanda grazie al numero crescente di applicazioni multimediali che si basano sulla trasmissione di immagini. Appare quindi fortemente desiderabile l'adozione di efficienti tecniche per la codifica di sorgente.

La codifica di sorgente si occupa di ridurre la quantità di dati con cui rappresentare l'informazione. Questo obiettivo si può realizzare identificando la ridondanza presente nei dati, cioè quella parte "superflua" che può essere eliminata causando nessuna o poca degradazione della qualità *ridondanza statistica*. Scoprire la ridondanza significa di fatto individuare le strutture nascoste nei dati che, una volta descritte attraverso un appropriato modello, possono essere usate efficientemente nel processo di codifica per effettuare compressione. In genere, questa è la fase più delicata dal momento che non è sempre possibile determinare facilmente il tipo di ridondanza. Spesso i dati devono essere elaborati attraverso sofisticati metodi di trasformazione prima di riuscire ad individuarne le strutture; altre volte, invece, è facile riconoscerle nei dati originari, come nel caso di testi o messaggi.

Uno dei primi a sfruttare la ridondanza (e quindi la struttura) della lingua inglese per effettuare codifica, è stato Samuel Morse, che nella metà del diciannovesimo secolo, ha sviluppato un codice per ridurre il tempo medio necessario a trasmettere un messaggio attraverso un telegrafo. Morse notò che alcune lettere dell'alfabeto si presentano più spesso di altre, e quindi, assegnò sequenze di punti e linee più brevi alle lettere che si presentavano più frequentemente, come $e\,(\cdot)$ ed $a\,(\cdot -)$, e più lunghe a quelle che si presentavano meno frequentemente, come $q\,(- - \cdot -)$ e $j\,(\cdot - - -)$. Questa stessa semplice idea è alla base di algoritmi attualmente usati per la compressione di file di dati (codifica di Huffmann). Anziché far riferimento alle lettere si può pensare all'occorrenza delle parole, così come accade nel codice Braille, sviluppato sempre nello stesso periodo e che permette una riduzione del venti per cento dello spazio occupato. La diversa frequenza di occorrenza delle lettere dell'alfabeto, o delle parole del vocabolario, rappresenta una struttura della sorgente d'informazione, la cui conoscenza rende possibile effettuare la compressione. In altri termini, nella sorgente esiste una certa regolarità, o *ridondanza statistica*, che permette di ottenere una sua rappresentazione meno dispendiosa. Questa appena descritta non è però l'unico tipo di struttura che esiste nei dati.

Un altro modo per risparmiare risorse di codifica è quello di rinunciare del tutto o in parte a rappresentare l'informazione che per l'utente sarebbe comunque irrilevante, cioè eliminare dalla sorgente la *ridondanza psicofisica*. In questo modo si sfrutta la conoscenza dell'utente finale che può avere una capacità risolutiva limitata, come nel caso in cui il destinatario dell'informazione sia un essere umano che ha limitata sensibilità visiva e uditiva. Per esempio, la sensibilità dell'occhio è ridotta in corrispondenza delle regioni periferiche dello spettro di un'immagine, cioè laddove sono elevati i valori della frequenza spaziale. Poichè frequenze spaziali elevate corrispondono alla rappresentazione di dettagli fini, l'osservatore non riesce a distinguerli. Un algoritmo, basato su criteri percettivi, può eliminare l'informazione irrilevante per il destinatario ultimo dell'informazione.

E' possibile individuare due diversi tipi di codifica di sorgente con obiettivi ed applicazioni distinte, la compattazione e la compressione dati. La prima sfrutta solo la ridondanza statistica e non prevede alcuna perdita d'informazione (codifica "lossless"), mentre la seconda trae vantaggio anche dalla ridondanza psicofisica e consente perdita d'informazione (codifica "lossy"). In entrambi i casi possiamo far riferimento allo schema di figura 1, nel quale sono evidenziati il codificatore, che a partire dall'immagine $x(m,n)$ genera la sua rappresentazione compatta in termini di flusso binario, e il decodificatore, che a partire dai dati compressi [1] ricostruisce

[1] supporremo sempre che il canale non introduca rumore per cui il flusso di bit in uscita al codificatore è identico a quello in ingresso al decodificatore.

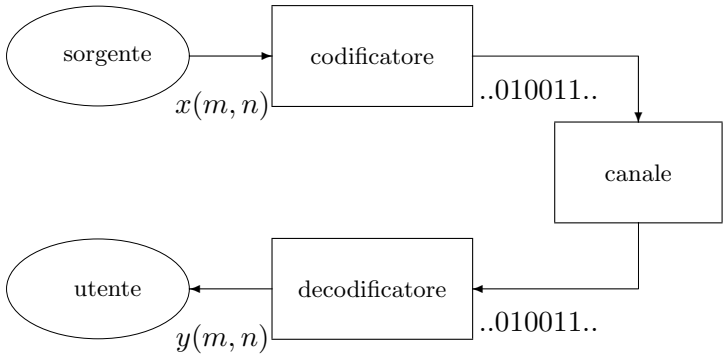


Figura 1: Diagramma a blocchi di un generico schema di compressione

esattamente il segnale originario ($y(m,n) \equiv x(m,n)$) o una sua approssimazione ($\widehat{x}(m,n)$).

Se si vogliono confrontare le prestazioni di tecniche differenti di compattazione dati si usa il *tasso di codifica* R, pari al numero medio di bit necessario a codificare un simbolo di sorgente. Dato che nella compressione si rinuncia alla possibilità di una perfetta ricostruzione dei dati originari è neceessario introdurre una misura di qualità dei dati ricostruiti, poichè bisogna individuare non solo di quanto si riduce l'informazione, ma anche qual è la degradazione causata, cioè specificare sia il tasso di codifica R che il livello di distorsione D introdotto. Si definiscono in questo modo le cosiddette *curve tasso-distorsione*, il cui andamento mostra come varia D al crescere di R. E' facile intuire che gli obiettivi di un'elevata efficienza di codifica e di una buona qualità del segnale siano contrastanti, per cui le tecniche di compressione usate dovranno essere tali da fornire il minor grado di distorsione per un fissato numero medio di bit per simbolo o, data la qualità di ricostruzione, il minor tasso di codifica possibile.

Nel seguito ci occuperemo esclusivamente delle tecniche di compressione, sebbene la compattazione (detta anche *codifica entropica*) venga applicata solitamente al flusso di dati già compressi allo scopo di ridurre ulteriormente i simboli da trasmettere senza introdurre perdite.

2 La compressione dati

La prospettiva di perdere informazioni genera forti resistenze verso la compressione dati, soprattutto in particolari ambienti, come la comunità medica o quella geologica. D'altra parte una perdita di informazione è inevitabile e implicitamente e tranquillamente accettata ogni volta che si quantizza un segnale per portarlo in forma numerica. In effetti, nel momento in cui si effettua una compressione si scambia solo un'ulteriore marginale degradazione del segnale con una maggiore efficienza di codifica, ed il vero nodo da sciogliere non è tanto *se* accettare una tale degradazione (che esiste comunque), ma solo *fino a che punto*. Ad esempio, per segnali destinati all'intrattenimento, tipo voce, immagini, video, è certamente lecito rimuovere, non solo la ridondanza statistica, ma anche quella psicofisica, poichè tale operazione, pur essendo a rigore irreversibile, risulta del tutto trasparente all'utente.

Per poter valutare le prestazioni di una tecnica di compressione lossy è importante definire una misura della degradazione introdotta; la più comune misura di distorsione è rappresentata dall'errore quadratico medio MSE (Mean Squared Error), definito come:

$$\text{MSE} = E[(X - \widehat{X})^2]$$

dove X è la variabile aleatoria che rappresenta il generico campione in ingresso, $\widehat{X}$ il valore che lo rappresenta dopo la decodifica ed $E[\cdot]$ indica l'operazione di media statistica. Quando le caratteristiche della sorgente non sono note, ma se ne può ipotizzare l'ergodicità, è lecito sostituire l'operazione di media statistica con quella di media temporale:

$$\text{MSE} = \frac{1}{N}\sum_{n=0}^{N-1}(x(n) - \widehat{x}(n))^2$$

dove gli $x(n),\ n = 1, \ldots, N$ sono i campioni estratti dalla sorgente e gli $\widehat{x}(n)$ i corrispondenti valori dopo la decodifica. Estendendo la definizione ad un immagine $x(m, n)$ di dimensioni $M \times N$, tale quantità si ottiene calcolando medie spaziali:

$$\text{MSE} = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(x(m,n) - \widehat{x}(m,n))^2 \tag{1}$$

con $\widehat{x}(m, n)$ immagine compressa. Per ottenere una misura che tenga conto della dinamica dei dati da codificare, e quindi risulti di più semplice interpretazione, si introduce il rapporto segnale-rumore (SNR), definito come il rapporto tra la potenza del segnale e quella del rumore in dB:

$$\text{SNR} = 10\log_{10}\frac{E[(X - E(X)]^2}{MSE} = 10\log_{10}\frac{\text{VAR}}{\text{MSE}}$$

dove $E(X)$ e $\text{VAR}(X)$ indicano rispettivamente media e varianza di X. Estendendo ancora una volta tale definizione alle immagini l'MSE è quello definito nella (2) mentre la varianza di un'immagine (supponendo di trovarci sempre in ipotesi di ergodicità) è:

$$\text{VAR} = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(x(m,n) - \mu_x)^2$$

con media μ_x:

$$\mu_x = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}x(m,n)$$

Spesso si usa anche il rapporto segnale-rumore di picco (PSNR), che per immagini codificate su 8 bit è definito come:

$$\text{PSNR} = 10\log_{10}\frac{255^2}{MSE}$$

Esiste una branca della teoria dell'informazione, la Rate-Distortion Theory, che si occupa proprio di trovare le prestazioni limite teoriche, in termini di curve $R(D)$ e $D(R)$ per assegnate sorgenti e misure di distorsioni. Anche per sorgenti piuttosto semplici, tuttavia, esistono pochi risultati in forma chiusa. Inoltre per molti modelli di immagine è difficile calcolare la curva limite perchè

manca un modello statistico esauriente del segnale. Infine, anche laddove questo risultasse possibile, i risultati non vanno considerati di valore assoluto perchè non è possibile definire un metodo di misura della distorsione universalmente valido, e si dovrà scegliere la misura che di volta in volta risulta più adatta all'applicazione.

2.1 Strategie di codifica

Le tecniche di codifica usate per la compressione dati si basano tutte sulla quantizzazione. Nonostante sia un'elaborazione molto semplice, il progetto del quantizzatore ha un impatto significativo sull'entità di compressione ottenuta: più i parametri del quantizzatore si adattano alle caratteristiche della sorgente migliori risulteranno le prestazioni del processo di codifica. Il quantizzatore può essere scelto in base alla distribuzione di probabilità con cui è stata modellata la sorgente oppure può essere progettato in base alla distribuzione empirica dei dati. Quando il segnale da codificare ha caratteristiche fortemente variabili nel tempo è conveniente usare un quantizzatore adattativo, in cui i parametri vengono cambiati in base alle statistiche del segnale.

Con la sola quantizzazione scalare non si sfrutta però la ridondanza presente nei dati. Per poter trarre vantaggio delle dipendenze statistiche tra i campioni di un segnale o tra i pixel di un'immagine si possono seguire due approcci:

- elaborare congiuntamente i simboli da quantizzare, per esempio progettando un *quantizzatore vettoriale* che però è caratterizzato da un'elevata complessità computazionale;
- decorrelare i dati, cioè eliminare solo le dipendenze di tipo lineare, attraverso una *predizione* o una *trasformazione*, e poi quantizzarli scalarmente.

Il secondo approccio, sebbene sub-ottimo, è molto semplice da implementare e permette di ottenere buoni risultati tanto da essere adottato nei più diffusi standard di compressione. Per comprendere l'importanza che il processo di decorrelazione assume nell'ambito della compressione si consideri il seguente esempio. Sia data la sequenza di simboli:

$$x_n = \{9, 11, 11, 11, 14, 13, 15, 17, 16, 17, 20, 21\} \qquad 1 \leq n \leq 12$$

supponendo di voler trasmettere o memorizzare tale sequenza in formato binario occorrerebbe utilizzare per ogni simbolo almeno 5 bit, per un totale di 60 bit. In realtà, da una semplice lettura dei dati, si nota che presentano un andamento abbastanza regolare, lentamente crescente, cioè è possibile riconoscere tra di essi una dipendenza lineare che può essere ad esempio modellata nel seguente modo:

$$\widehat{x}_n = n + 8$$

Se ora consideriamo la sequenza ottenuta prendendo in esame la differenza esistente tra i campioni omologhi delle due sequenze, ovvero la sequenza errore o residuo:

$$e_n \; = \; x_n - \widehat{x}_n = \{0, 1, 0, -1, 1, -1, 0, 1, -1, -1, 1, 1\}$$

notiamo che tale sequenza è costituita solo da tre simboli, pertanto saranno necessari solo 2 bit per rappresentare ogni elemento per un totale di 24 bit. Ovviamente il decodificatore dovrà conoscere il modello utilizzato e la sequenza errore per ricostruire il segnale originario. Si noti che e_n non presenta più la forte struttura che caratterizza i dati originari, anzi i campioni

appaiono fra loro incorrelati. In effetti, se esistesse una qualche residua dipendenza fra i dati, questo sarebbe indice che il modello usato non era il migliore possibile. La decorrelazione dei dati si presenta, quindi, come parametro indicativo di quanto efficientemente sia stata sfruttata la ridondanza presente nella sequenza di partenza. Maggiore è la decorrelazione dei campioni di uscita e più si è ridotta la quantità dei dati da trasmettere. Ovviamente è necessario individuare le strutture nascoste nei dati attraverso un adeguato modello e ciò non è sempre semplice.

Nel seguito parleremo in generale del processo di quantizzazione, quindi daremo qualche cenno alla quantizzazione predittiva per poi concentrarci sulla codifica basata su trasformata. Per la trattazione della quantizzazione faremo riferimento ad un solo simbolo x di sorgente (modellato come una variabile aleatoria X), essendo l'elaborazione scalare. Invece nel descrivere sia le tecniche di predizione che di trasformazione lavoreremo su una sequenza $x(n)$ o vettore $\mathbf{x}$, e se l'approccio è di tipo probabilistico su un vettore di variabili aleatorie $\mathbf{X}$. Infine, estenderemo le tecniche di interesse all'immagine $x(m, n)$.

3 La Quantizzazione

La quantizzazione scalare (SQ) è il processo di compressione "naturale" che subisce un segnale analogico quando viene digitalizzato; se applicata ad un segnale già in forma numerica permette di ridurre il numero di bit necessari alla codifica.

I possibili valori che può assumere il segnale in ingresso vengono raggruppati in un determinato numero di insiemi, chiamati *celle* o *regioni*, ognuno dei quali è caratterizzato da un singolo valore di uscita, detto *livello di restituzione*. Una volta individuato per ogni valore del segnale di ingresso la cella di appartenenza, si potrà associare ad esso il livello di restituzione relativo. La quantizzazione realizza, quindi, la corrispondenza:

$$\mathcal{Q}: \; x \in X \subseteq \mathbb{R} \; \rightarrow \; \mathcal{Q}(x) \in \mathcal{C} = \{y_1, y_2, \ldots, y_N\} \subset \mathbb{R} \tag{2}$$

ad ogni valore appartenente ad un sottoinsieme X dei numeri reali $\mathbb{R}$ viene associata una parola appartenente a $\mathcal{C}$, detto il codice del quantizzatore di cardinalità N. Di conseguenza, le regioni R_i, in cui è partizionato l'asse reale sono date da:

$$R_i \equiv \{x \in X : \mathcal{Q}(x) = y_i\} \tag{3}$$

Un quantizzatore è completamente descritto una volta specificati i livelli di uscita $\{y_i; \; i = 1 \ldots N\}$ e le corrispondenti regioni $\{R_i; \; i = 1 \ldots N\}$. Tipicamente il quantizzatore appena definito soddisfa le condizioni di *regolarità*:

1. ogni cella R_i è un intervallo (x_{i-1}, x_i);
2. $y_i \in (x_{i-1}, x_i)$;

Considereremo unicamente quantizzatori regolari, in tal caso i valori $\{x_i\}_{i=0}^{N}$ che definiscono le regioni sono detti *soglie di decisione* e l'ampiezza dell'i-esima cella, Δ_i è data da $x_i - x_{i-1}$. La quantizzazione può essere vista come l'effetto di due successive operazioni, una codifica e una decodifica (fig.2). Il codificatore realizza la corrispondenza

$$\mathcal{E}: \; x \in R_i \rightarrow \mathcal{E}(x) = i \in \mathcal{I} \tag{4}$$

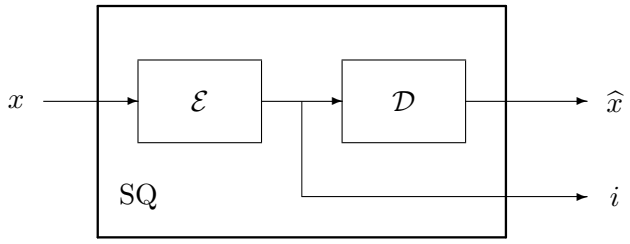


Figura 2: Quantizzatore scalare

dove $\mathcal{I} = \{1, 2, \ldots, N\}$, cioè per ogni valore di ingresso esso individua la cella di appartenenza e si limita a trasmettere l'indice corrispondente. Il numero di bit necessari a specificare il valore quantizzato, pari a $R = \log_2 N$, è detta *risoluzione* o *tasso di codifica*, e dà indicazioni sull'accuratezza con cui vengono rappresentati i dati. Il decodificatore è invece definito come

$$\mathcal{D} : \; i \in \mathcal{I} \rightarrow \mathcal{D}(i) = y_i \in \mathcal{C} \tag{5}$$

e associa ad ogni indice la parola codice corrispondente, fornita dalla tabella codice. Il quantizzatore è allora costituito dalla cascata del codificatore e del decodificatore:

$$\mathcal{Q}(x) = \mathcal{D}(\mathcal{E}(x)) = y_i \tag{6}$$

L'operazione realizzata dalla quantizzazione corrisponde proprio al concetto di approssimazione di un numero reale con una rappresentazione a precisione finita. E' evidente che, a causa della corrispondenza "molti a uno" realizzata dal codificatore, si ha un'inevitabile perdita di informazione, non recuperabile dal decodificatore che commette un errore in fase di ricostruzione. Una misura della distorsione causata, come già detto, può essere determinata dall'errore quadratico medio (MSE), per cui nell'ipotesi di modellare ogni campione del segnale in ingresso come una variabile aleatoria X con densità di probabilità $f_X(x)$, si ha:

$$\begin{aligned} D = \text{MSE} = E[(X - \widehat{X})^2] &= \int_{\mathbb{R}} (x - \widehat{x})^2 f_X(x)\, dx \\ &= \sum_{i=1}^{N} \int_{R_i} (x - y_i)^2 f_X(x)\, dx \end{aligned} \tag{7}$$

E' bene notare che l'errore che si commette per i valori di ampiezza appartenenti agli intervalli interni è, in generale, diverso da quello che si commette per quelli che appartengono ai due intervalli estremi. Infatti, l'errore di quantizzazione $e = x - \widehat{x}$ si mantiene limitato per gli intervalli interni (*rumore granulare*), mentre risulta essere di solito illimitato per gli intervalli estremi (*rumore di sovraccarico*).

3.1 Quantizzazione uniforme

Si parla di quantizzazione *uniforme* quando tutti gli intervalli hanno la stessa durata Δ e i livelli di restituzione (per gli intervalli interni) coincidono con i valori centrali di ogni cella:

$$y_i = \frac{x_{i-1} + x_i}{2}$$

così come mostrato in figura 3 nel caso in cui $N = 8$, un esempio della legge di quantizzazione relativa è presentato in figura 4. In figura 5, invece, si mostra un altro esempio di quantizzatore che prevede lo zero come livello di restituzione (e ha un numero dispari di livelli $N = 7$).

Un quantizzatore uniforme è facile da progettare ed è ottimo se anche i dati da quantizzare presentano una pdf uniforme. Nell'ipotesi allora che $X \sim U(-\frac{B}{2}, \frac{B}{2})$, valutiamo la distorsione al variare del tasso di codifica, usando la (8) risulta:

$$\begin{aligned} D &= \sum_{i=1}^{N} \int_{x_{i-1}}^{x_i} (x - y_i)^2 f_X(x)\, dx \\ &= \sum_{i=1}^{N} \int_{x_{i-1}}^{x_i} \frac{1}{B}(x - y_i)^2\, dx = \frac{1}{B} \sum_{i=1}^{N} \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} x^2\, dx \qquad (8) \\ &= \frac{N}{B} \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} x^2\, dx = \frac{N}{B} \cdot \frac{\Delta^3}{12} = \frac{\Delta^2}{12} \qquad (9) \end{aligned}$$

nella (9) è stato fatta la sostituzione $x \to x + y_i$. Poiché $\Delta = \frac{B}{N}$ è proprio l'ampiezza delle celle di quantizzazione, si ottiene:

$$D = \frac{1}{12} B^2 N^{-2} = \frac{1}{12} B^2 2^{-2R} \qquad (10)$$

dove si è ricordato che $R = \log_2 N$. Allora quando si ha a che fare con una variabile aleatoria uniforme quantizzata a sua volta in modo uniforme, ogni volta che si incrementa il tasso di codifica di un bit la distorsione si riduce di un fattore quattro. Si noti che, se non si spendono risorse ($R = 0$), la migliore approssimazione del segnale è la sua media: $\hat{x} = \mu_X$, in questo caso si commette un errore quadratico medio che coincide con la varianza del segnale. Infatti:

$$\begin{aligned} \sigma_X^2 = E[(X - \mu_X)^2] &= \int_{\mathbb{R}} (x - \widehat{x})^2 f_X(x)\, dx \\ &= \int_{-B/2}^{B/2} \frac{1}{B}\, x^2\, dx = \frac{B^2}{12} \end{aligned}$$

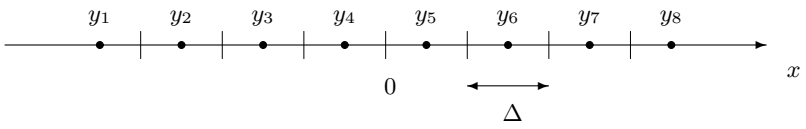


Figura 3: Suddivisione dell'asse reale in 8 intervalli di quantizzazione.

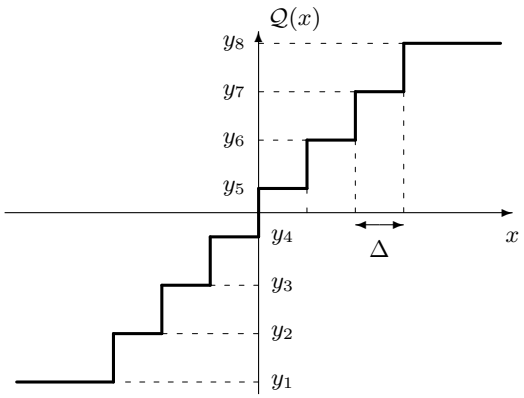


Figura 4: Esempio di quantizzatore uniforme.

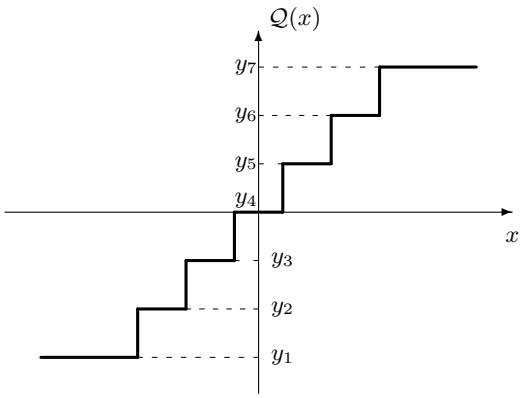


Figura 5: Esempio di quantizzatore uniforme con restituzione dello zero.

per cui

$$D = \sigma_X^2 \, 2^{-2R} \tag{11}$$

Per mostrare gli effetti della quantizzazione su un segnale reale, consideriamo un'immagine in formato numerico che può assumere 256 possibili livelli di grigio. Quantizziamo poi l'immagine con un quantizzatore uniforme con un numero via via crescente di livelli di restituzione e calcoliamo l'MSE costruendo la la curva tasso-distorsione relativa (figura 6). E' interessante notare che se visualizziamo le immagini ricostruite a diversi tassi (figura 7), l'errore di quantizzazione risulta evidente per $N = 2, 4, 16$, mentre è quasi impercettibile per $N = 64$. Ciò è dovuto alle limitate capacità percettive dell'occhio umano, che riesce con difficoltà a distinguere due livelli di grigio molto vicini tra loro. Pertanto, se è vero che la quantizzazione produce un errore sui dati, è anche vero che molto spesso questo errore non è percepito dall'utente finale che usufruisce

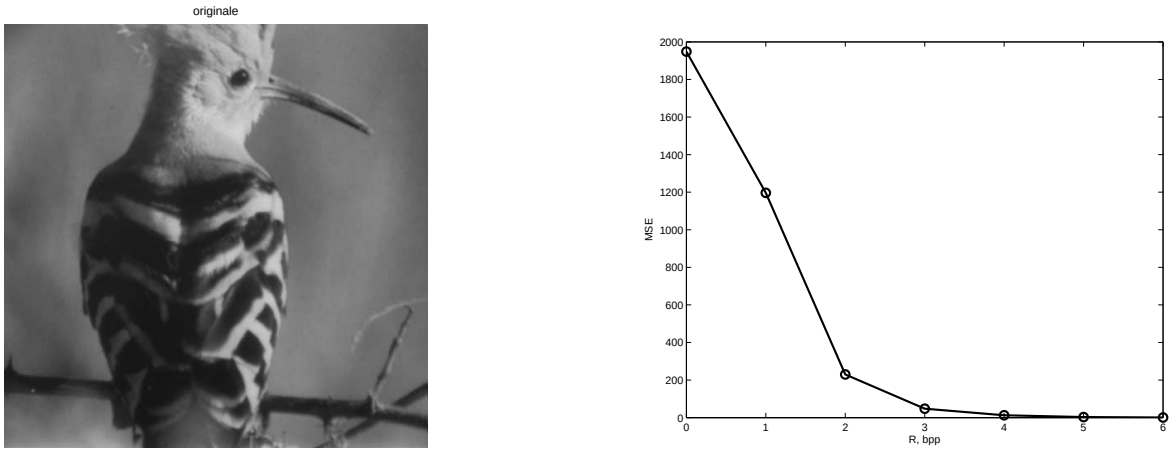


Figura 6: Curva tasso-distorsione relativa alla quantizzazione dell'immagine a sinistra.

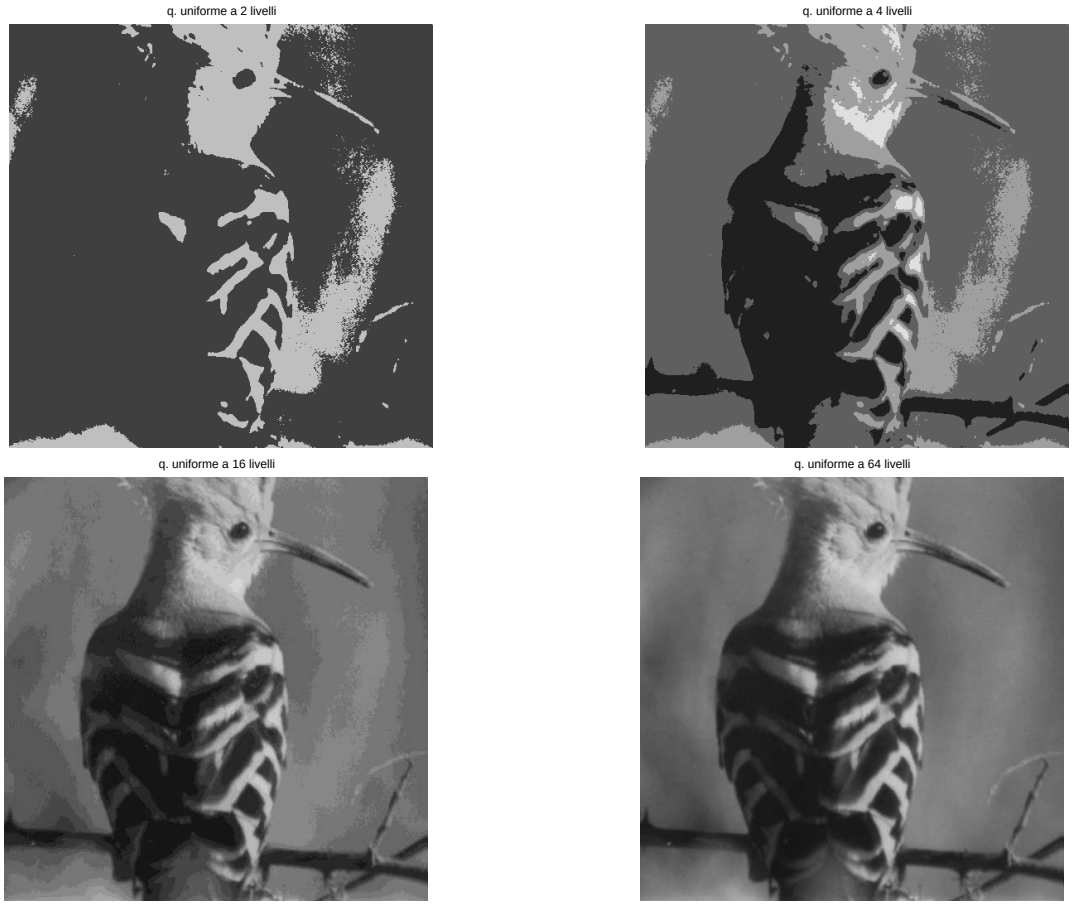


Figura 7: Immagini quantizzate con 2, 4, 16 e 64 livelli di restituzione.

dell'informazione (sia essa un'immagine, un segnale video o audio), soprattutto se il quantizzatore è progettato in modo opportuno. E' infatti possibile progettare un quantizzatore in modo che si adatti alle caratteristiche dei dati da elaborare e a quelle dell'utente finale, minimizzando così l'errore che necessariamente si commette con questa operazione.

3.2 Quantizzazione non uniforme

Se la distribuzione dei dati non è uniforme non è conveniente usare sempre lo stesso passo di quantizzazione. Per esempio, se con elevata probabilità i valori assunti dal segnale si concentrano intorno all'origine, conviene usare passi di quantizzazione stretti intorno allo zero e ampi lontano dall'origine, così come mostrato in figura 8.

Il progetto del quantizzatore ottimo per una assegnata distribuzione dei dati consiste proprio, una volta assegnato il tasso del codice, nel scegliere le partizioni dell'asse reale e i relativi livelli di restituzione che minimizzino l'errore quadratico medio. Se si trascura il caso di quantizzazione ad alta risoluzione, dove $R \gg 1$, questo problema non può essere risolto in forma chiusa. La mancanza di una soluzione diretta è dovuta alla difficoltà di trattare con la natura altamente non lineare della quantizzazione, tuttavia esistono due condizioni necessarie, ma non sufficienti, per l'ottimalità.

Per determinare la prima delle due condizioni si fissa il decodificatore (cioè il codice) e si sceglie la migliore partizione dell'asse reale che minimizza la misura di distorsione scelta; nel caso dell'errore quadratico medio risulta:

$$R_i = \{x : |x - y_i| \leq |x - y_j| \quad \forall \ j \neq i\} \qquad i = 1, 2, \ldots, N \tag{12}$$

La cella i-esima è quindi costituita da tutti i punti più vicini a y_i che ad ogni altra parola codice (*nearest neighbor condition*). Per la seconda condizione si fissa il codificatore (cioè la partizione) e si ottimizza il decodificatore minimizzando la distorsione, e si può dimostrare che si ottiene:

$$y_i = \int_{R_i} x f_{X/R_i}(x) dx = \frac{\int_{R_i} x f_X(x) dx}{\int_{R_i} f_X(x) dx} \qquad i = 1, 2, \ldots, N \tag{13}$$

Il livello di uscita ottimo coincide cioè con il baricentro della distribuzione di probabilità, detto anche centroide, in ciascuna delle regioni fissate dal codice (*centroid condition*). Note le due condizioni necessarie per il progetto di un quantizzatore ottimo, esiste un algoritmo molto semplice dovuto a Lloyd in cui iterativamente il codificatore e il decodificatore vengono ottimizzati fino a raggiungere la convergenza per entrambi.

3.3 Quantizzazione ad alta risoluzione

Nell'ipotesi in cui il tasso di codifica sia sufficientemente elevato ($R \gg 1$) si può dimostrare che la relazione tasso-distorione del quantizzatore ottimo è:

$$D = h_X \, \sigma_X^2 \, 2^{-2R} \tag{14}$$

Si noti come sia molto simile alla (11) (valida solo per quantizzatori uniformi); l'unica differenza è la presenza di h_x detto *fattore di forma*, che dipende solo dal tipo di pdf di X, $f_X(x)$, ma non

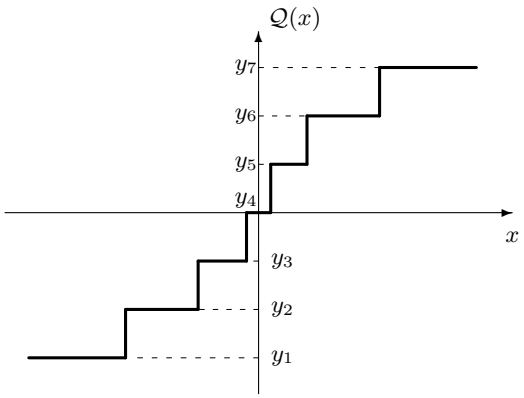


Figura 8: Esempio di quantizzatore non uniforme.

dalla sua varianza, attraverso la relazione

$$h_X = \frac{1}{12}\left[\int_{-\infty}^{+\infty} f_{\widetilde{X}}^{1/3}(x)dx\right]^3 \tag{15}$$

con $\widetilde{X} = X/\sigma_X$, e vale 1 per distribuzione uniforme, e $\sqrt{3}\pi/2$ per distribuzione gaussiana.

4 Codifica predittiva

Abbiamo già accennato al fatto che l'elaborazione congiunta di simboli di sorgente permette di migliorare le prestazioni complessive dello schema di codifica, dal momento che si sfrutta la dipendenza statistica dei dati. Un modo molto semplice per raggiungere questo obiettivo è quello di realizzare una codifica predittiva seguita da una quantizzazione scalare. Per comprendere l'idea alla base di questo approccio mostriamo un esempio, considerando la seguente sequenza di numeri:

27 28 29 28 26 27 29 28 30 32 34 36 38

Notiamo che questi valori sono abbastanza simili tra loro, è presente cioè un certo livello di correlazione (dipendenza lineare) tra i dati di cui si può trarre vantaggio. Supponiamo allora di inviare il primo valore e poi consideriamo la differenza tra due campioni vicini:

27 1 1 −1 −2 1 2 −1 2 2 2 2 2

Osserviamo come adesso il numero di valori distinti tra loro è diminuito e soprattutto la dinamica dei dati (a parte il valore iniziale) è stata ridotta sensibilmente. Questo significa che a parità di risorse spese in fase di quantizzazione la distorsione diminuisce, o equivalentemente è possibile usare un numero inferiore di bit fissato il livello di distorsione. L'esempio mostrato utilizza una forma di predizione molto semplice in cui il campione attuale viene stimato con quello precedente e si trasmette la differenza tra i due, detto *errore di predizione*. Formalizzando il discorso, detta

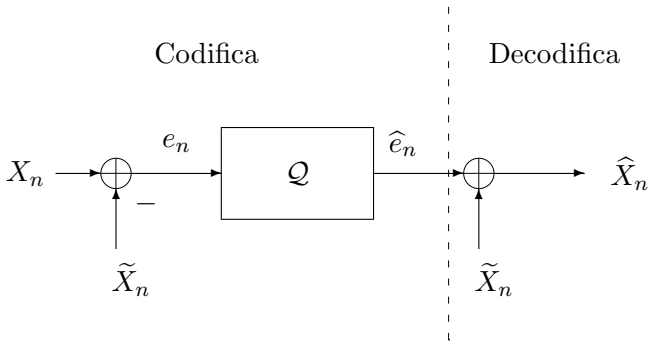


Figura 9: Codifica predittiva.

X_n la variabile aleatoria associata all'n-esimo simbolo di sorgente e indicata con $\widetilde{X}_n$ la sua stima ciò che viene trasmesso è:

$$e_n = X_n - \widetilde{X}_n$$

Nell'esempio si è posto $\widetilde{X}_n = X_{n-1}$. Ovviamente si possono considerare modelli più complessi che portano in conto una combinazione lineare di simboli passati per stimare quello attuale. Più in generale si sfrutta cioè la conoscenza dei valori passati per predire quello corrente:

$$\widetilde{X}_n = f(X(n-1), X(n-2), \ldots, X(n-M)) \tag{16}$$

Si produce una stima di X_n a partire da M campioni precedenti, che nel caso lineare diventa:

$$\widetilde{X}_n = \sum_{i=1}^{M} a_i X(n-i) \tag{17}$$

dove $\{a_i\}$ sono i coefficienti del predittore ottimo. In questo modo se i dati presentano elevata correlazione, l'incertezza associata a nuovi simboli emessi dalla sorgente risulterà minore e si guadagna in termini di compressione.

Ovviamente e_n va quantizzato per cui sul canale viene trasmesso $\widehat{e}_n$ e quindi in fase di decodifica bisogna sommare nuovamente la sua stima per riottenere una versione approssimata del simbolo iniziale $\widehat{X}_n$. Lo schema a blocchi corrispondente è mostrato in figura 9.

La codifica predittiva è usata per ridurre la ridondanza temporale del segnale video, infatti è presente una forte correlazione tra i campioni di immagini successive. La predizione può essere semplicemente l'immagine precedente o una sua elaborazione ottenuta con la tecnica di compensazione del moto.

Esempio: *Confronto prestazioni.* Supponiamo che risulti $X_n \sim \mathcal{N}(0, \sigma_X^2)$, e ipotizziamo che campioni vicini siano fortemente correlati, per cui $E[X_n\, X_{n-1}] = \rho\, \sigma_X^2$, con $\rho = 0.9$ coefficiente di correlazione[2]. Supponiamo poi che la stima sia pari proprio al campione precedente per cui

[2]Si ricorda che il coefficiente di correlazione tra due variabili aleatorie, X e Y, si definisce come

$$\rho = \frac{\mathrm{COV}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

$\widetilde{X}_n = X_{n-1}$ e l'errore di predizione è $e_n = X_n - \widetilde{X}_n = X_n - X_{n-1}$. Ci proponiamo di calcolare la distorsione D e confrontarla con quello che si ottiene quando si effettua predizione D_p. Abbiamo già detto che la distorsione si definisce come:

$$D = \text{MSE} = E[(X_n - \widehat{X}_n)^2]$$

D'altra parte, essendo valida la (14) con $h_X = \frac{\pi\sqrt{3}}{2}$, risulta:

$$D = h_X\, \sigma_X^2\, 2^{-2R} = \frac{\pi\sqrt{3}}{2}\, \sigma_X^2\, 2^{-2R} \tag{18}$$

Per un quantizzatore predittivo, invece, è valida la (21) per cui:

$$D_p = E[(e_n - \widehat{e}_n)^2] = h_e\, \sigma_e^2\, 2^{-2R} = \frac{\pi\sqrt{3}}{2}\, \sigma_e^2\, 2^{-2R} \tag{19}$$

dove si è notato che e_n è ancora gaussiana in quanto somma di gaussiane e quindi $h_e = h_X$. Si noti come le due relazioni, la (18) e la (19), differiscano di un fattore pari a σ_X^2/σ_e^2, che viene definito *guadagno di predizione* G_p:

$$G_p \stackrel{\triangle}{=} \frac{\sigma_X^2}{\sigma_e^2}$$

se $G_p > 1$ allora $\sigma_e^2 < \sigma_X^2$ per cui con la predizione si guadagna, perché la distorsione risulterà minore, a parità di risorse spese. Inoltre risulta:

$$\begin{aligned}\sigma_e^2 = E[e_n^2] &= E[(X_n - X_{n-1})^2] \\ &= E[X_n^2] + E[X_{n-1}^2] - 2\,E[X_n\, X_{n-1}] \\ &= \sigma_X^2 + \sigma_X^2 - 2\,\rho\sigma_X^2 = 2\sigma_X^2(1-\rho)\end{aligned}$$

In conclusione

$$D_p = \pi\sqrt{3}\sigma_X^2(1-\rho)\, 2^{-2R} \tag{20}$$

quindi il guadagno di predizione è:

$$G_p = \frac{1}{2(1-\rho)}$$

e per $\rho = 0.9$ è pari a 5, quindi fissato R, cioè a parità di risorse, la distorsione diminuisce nel caso di codifica predittiva.

Esempio: *Confronto istogrammi.* In figura 10 è mostrata l'immagine originale con il relativo istogramma. Si può notare che i valori assunti dali livelli di grigio variano quasi uniformemente in tutto il range (da 0 a 255). Chiaramente per rappresentare questi valori sono necessari perfettamente 8 bit per pixel, e per effettuare compressione una quantizzazione con 5 bit per pixel garantisce la trasparenza percettiva. Supponiamo poi di considerare una codifica predittiva in cui la predizione è pari proprio al pixel precedente (lungo le righe), e visualizziamo il relativo istogramma (fig.11). Si può notare come più del 99 % dei pixel si trova nel range $[-50, 50]$, con elevata probabilità che cadano intorno allo zero (pdf di tipo Laplace), per cui quantizzando non uniformemente nell'intervallo $[-50, 50]$ si riduce il numero di bit/pixel necessari per la codifica.

Risulta $|\rho| \leq 1$ con uguaglianza se e solo se $Y = aX$, con a costante. Inoltre, due variabili aleatorie si dicono incorrelate se $\rho = 0$.

4.1 Codificatore/decodificatore

Se è vero che sono necessari un numero inferiore di bit per codificare le differenze piuttosto che i valori originali, nello schema di figura 9 abbiamo trascurato il fatto che in decodifica i valori originali X_n non sono disponibili per cui la stima sarà determinata dai valori quantizzati $\widehat{X}_n$. Questo significa che considerando sempre come stima il campione precedente in decodifica non possiamo calcolare la stima come $\widetilde{X}_n = X_{n-1}$, bensì $\widetilde{X}_{qn} = \widehat{X}_{n-1}$. Quindi gli schemi del codificatore e decodificatore sono in realtà quelli mostrati in figura 12. Il fatto che la stima in decodifica sia valutata attraverso i valori quantizzati può causare dei problemi, dato che la predizione basata sui dati può discostarsi di molto da quella basata sui dati quantizzati. Per comprendere il problema facciamo ancora una volta un esempio e consideriamo la sequenza x_n:

$$6.2 \;\; 9.7 \;\; 13.2 \;\; 5.9 \;\; 8 \;\; 7.4 \;\; 4.2 \;\; 1.8$$

La sequenza differenza e_n risulta essere:

$$6.2 \;\; 3.5 \;\; 3.5 \;\; -7.3 \;\; 2.1 \;\; -0.6 \;\; -3.2 \;\; -2.4$$

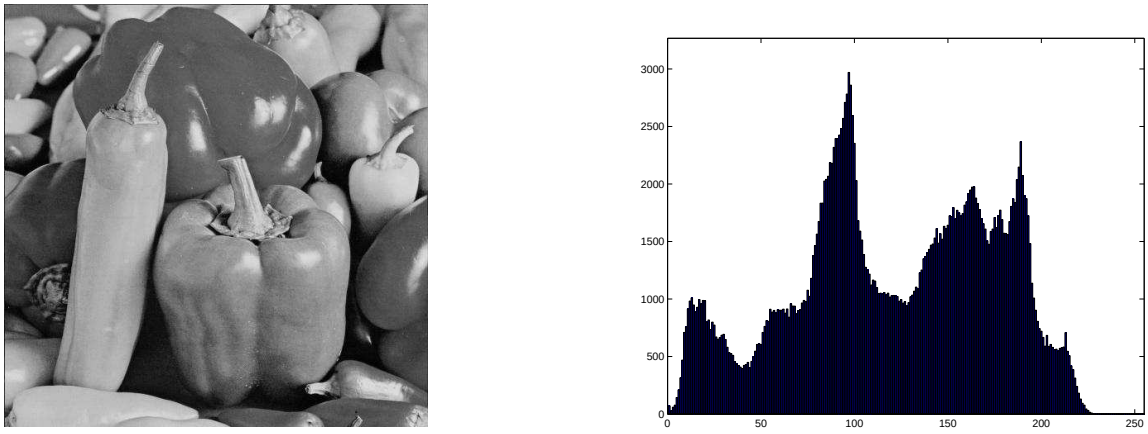


Figura 10: Immagine originale e relativo istogramma.

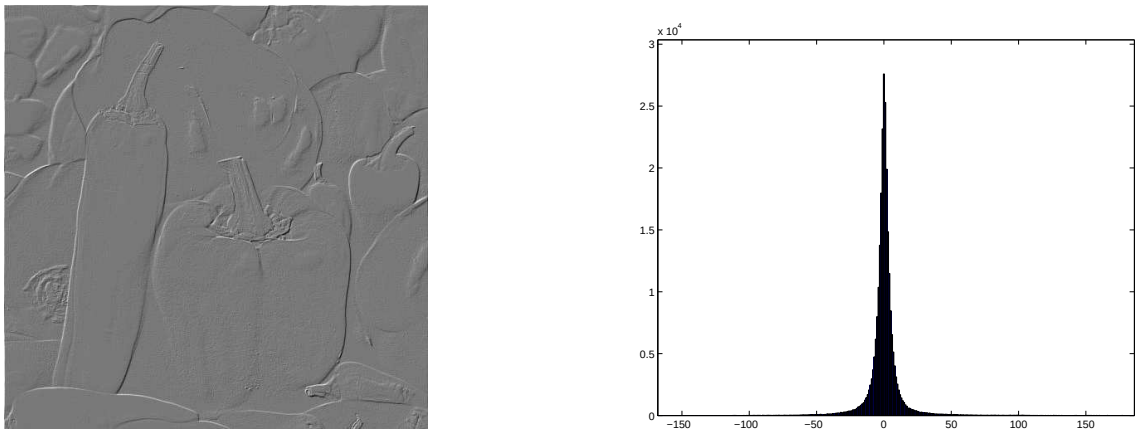


Figura 11: Immagine differenza e relativo istogramma.

Se questi dati non fossero quantizzati non avremmo problemi a recuperare la sequenza originale. Nell'ipotesi di avere a disposizione un quantizzatore uniforme con i 7 livelli di restituzione: -6, -4, -2, 0, 2, 4, 6, la sequenza quantizzata $\widehat{e}_n$ risulta essere:

$$6 \;\; 4 \;\; 4 \;\; -6 \;\; 2 \;\; 0 \;\; -4 \;\; -2$$

A questo punto in decodifica ricaviamo i valori ricostruiti come $\widehat{x}_n = \widehat{e}_n + \widetilde{x}_n = \widehat{e}_n + \widehat{x}_{n-1}$:

$$6 \;\; 10 \;\; 14 \;\; 8 \;\; 10 \;\; 10 \;\; 6 \;\; 4$$

Adesso possiamo determinare l'errore che commettiamo in ricostruzione pari a $x_n - \widehat{x}_n$:

$$0.2 \;\; -0.3 \;\; -0.8 \;\; -2.1 \;\; -2 \;\; -2.6 \;\; -1.8 \;\; -2.2$$

Si noti come inizialmente il valore assoluto dell'errore sia abbastanza piccolo, ma poi cominci a crescere. Ciò che accade di fatto è che l'errore dovuto alla quantizzazione si accumula man mano che il processo di predizione va avanti, portando il decodificatore a produrre valori sempre più diversi da quelli desiderati (disallineamento o deriva del decodificatore). In realtà, teoricamente se l'errore di quantizzazione è a media nulla gli errori dovrebbero cancellarsi se osserviamo evolvere il processo per un tempo sufficientemente lungo. Tuttavia, a causa della precisione finita del calcolatore ciò non accade.

In effetti, come è possibile notare dalla figura 12 codificatore e decodificatore operano in modo diverso, perché in fase di codifica la differenza è generata basandosi sui valori originali, mentre in decodifica con quelli quantizzati. Allora il problema si risolve se anche il codificatore viene forzato ad effettuare la stima con i valori quantizzati (figura 13). Predire X_n dai valori quantizzati passati, anziché da quelli originali, permette di generare $\widetilde{X}_{qn}$ sia in fase di codifica che di decodifica allo stesso modo (si ricordi che comunque è necessario conoscere il valore iniziale in decodifica). In questo modo non ci sarà accumulo del rumore di quantizzazione, e risulterà $\sigma_e^2 < \sigma_X^2$, quindi il guadagno di predizione G_p risulterà maggiore di uno e si avrà un miglioramento complessivo delle prestazioni.

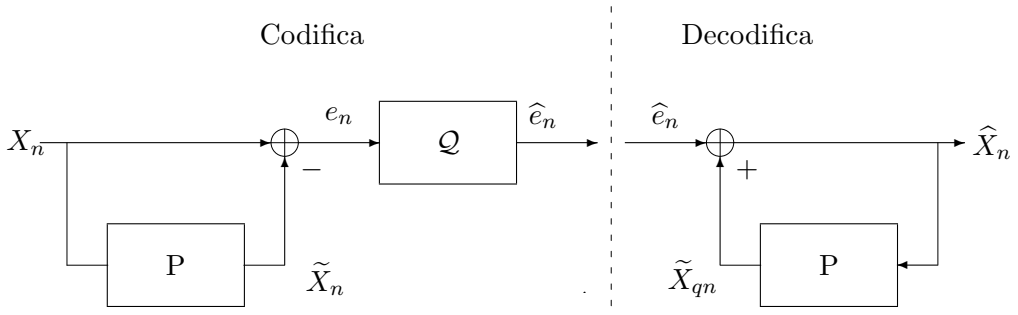


Figura 12: Quantizzatore predittivo

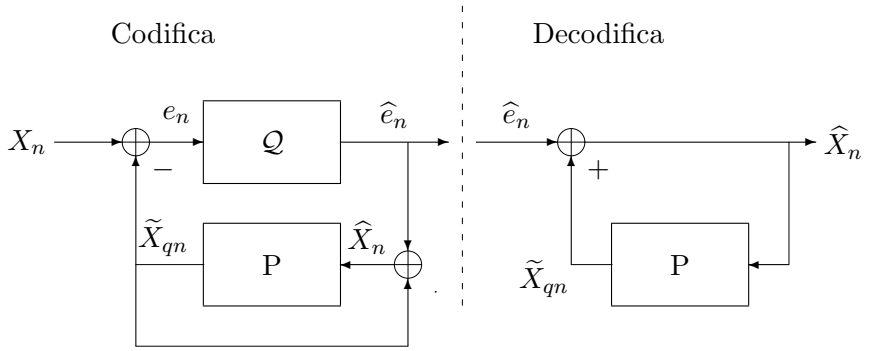


Figura 13: Quantizzatore predittivo

5 Codifica con trasformata

Un altro modo per sfruttare la dipendenza esistente tra i dati è quello di usare una trasformazione prima della quantizzazione. Il vantaggio che si ottiene nell'elaborare il segnale nel dominio trasformato risiede nel fatto che le proprietà statistiche dei nuovi coefficienti sono in genere diverse da quelle dei dati originari. L'uso di una trasformata ha essenzialmente due obiettivi (che in realtà sono legati l'un l'altro):

- fare sì che i coefficienti nel nuovo dominio siano decorrelati, o comunque "più indipendenti" di quelli del vettore originario;
- concentrare l'energia del segnale in un numero di coefficienti molto più piccolo rispetto a quello dei campioni del blocco in ingresso.

La compattazione dell'energia, in particolare, permette di realizzare una più efficiente quantizzazione scalare nel dominio trasformato. La diversa distribuzione di energia nel nuovo dominio conferisce infatti ai vari coefficienti importanza diversa e permette di ottimizzare le prestazioni globali del codificatore assegnando un numero diverso di bit ai coefficienti da quantizzare.

Inoltre, spesso il dominio trasformato risulta più appropriato per effettuare una quantizzazione che usa criteri percettivi, poiché alcuni coefficienti, anche se di energia non trascurabile, possono essere scartati perchè non alterano la qualità percepita dall'utente. Questa considerazione rende adeguato l'uso di una codifica con trasformata che separa le diverse componenti frequenziali, perchè in tal modo si possono trascurare quelle relative alle alte frequenze cui compete una ridotta sensibilità da parte dell'utente e spesso anche uno scarso contenuto energetico.

Il generico schema di compressione che fa uso di una trasformazione seguita dalla quantizzazione prevede come ultima operazione la compattazione, allo scopo di ridurre ulteriormente il tasso senza perdita di informazione (fig.14). Nello sviluppo dello schema di codifica con trasformata trascureremo il blocco che effettua la compattazione e focalizzeremo l'attenzione sulla scelta della trasformata e sul processo di quantizzazione. Inoltre limiteremo l'attenzione alle trasformate di lineari, che sono estremamente semplici da studiare.

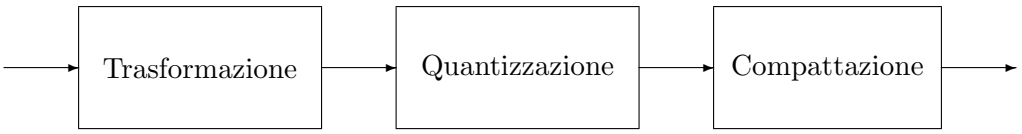


Figura 14: Schema di codifica

5.1 Le trasformate lineari

Detto $x(n)$ un segnale tempo discreto di lunghezza N, la trasformata *diretta* è definita nel seguente modo:

$$y(k) = \sum_{n=0}^{N-1} a_{k,n}\, x(n), \qquad 0 \le k \le N-1 \tag{21}$$

Per recuperare il segnale originario, è necessario realizzare la trasformata *inversa*, vale a dire:

$$x(n) = \sum_{k=0}^{N-1} b_{n,k}\, y(k), \qquad 0 \le n \le N-1 \tag{22}$$

Le due relazioni possono essere scritte in forma più compatta come:

$$\begin{cases} \mathbf{y} &= \mathbf{A}\mathbf{x} \\ \mathbf{x} &= \mathbf{B}\mathbf{y} \end{cases}$$

dove $\mathbf{A}$ e $\mathbf{B}$ sono matrici $N \times N$ per cui risulta $[\mathbf{A}]_{i,j} = a_{i,j}$ e $[\mathbf{B}]_{i,j} = b_{i,j}$. Chiaramente le due matrici sono una l'inversa dell'altra, cioè risulta: $\mathbf{AB} = \mathbf{BA} = \mathbf{I}$, dove $\mathbf{I}$ è la matrice identica. Trasformate di particolare interesse sono quelle che godono della proprietà di *ortogonalità*, per le quali la matrice inversa coincide con quella diretta coniugata e trasposta

$$\mathbf{B} = \mathbf{A}^{-1} = \mathbf{A}^{*T} \tag{23}$$

Nel seguito considereremo sempre trasformate reali e ortonormali per cui

$$\mathbf{A}^{-1}\mathbf{A} = \mathbf{A}^{T}\mathbf{A} = \mathbf{I}$$

in tal caso la matrice è anche detta *unitaria*. Le trasformazioni ortogonali assumono un ruolo fondamentale, poiché corrispondono a semplici rotazioni nello spazio N-dimensionale e quindi conservano l'energia del vettore nel dominio trasformato. Infatti risulta:

$$\begin{aligned} E_y = \sum_{k=0}^{N-1} y(k)^2 = \|\mathbf{y}\|^2 = \mathbf{y}^T\mathbf{y} = \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} \\ = \mathbf{x}^T\mathbf{A}^{-1}\mathbf{A}\mathbf{x} = \mathbf{x}^T\mathbf{x} = \|\mathbf{x}\|^2 = \sum_{n=0}^{N-1} x(n)^2 = E_x \end{aligned} \tag{24}$$

Esempio: Consideriamo un esempio molto semplice in cui $\mathbf{x}$ è un vettore costituito dall'informazione su altezza e peso di un individuo. Rappresentiamo questa coppia di informazioni su di un grafico così come mostrato in figura 15 e notiamo come tali valori tendano ad addensarsi intorno alla retta $y = \frac{5}{2}x$. I dati sono evidentemente fortemente correlati: più grande è il peso di un individuo maggiore è la probabilità che sia alto e viceversa. Proviamo adesso a ruotare questo insieme di dati attraverso la matrice di rotazione $\mathbf{A}$:

$$\mathbf{A} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix}$$

dove $\phi = \arctan(5/2)$ è l'angolo compreso tra l'asse delle ascisse e la retta $y = \frac{5}{2}x$. Realizziamo cioè la trasformazione: $\mathbf{y} = \mathbf{Ax}$. Come si può vedere in figura 15 la rotazione ha compattato l'energia nella prima componente y_1, mentre la seconda componente y_2 assume valori prossimi a zero (non c'è più correlazione tra i dati).

Se ponessimo a zero questa seconda componente (riducendo così alla metà i dati da codificare) e realizzassimo la trasformazione inversa per riottenere $\mathbf{x}$ otterremmo dei dati molto prossimi a quelli originali. Tutto ciò è reso possibile dall'aver individuato una trasformazione in grado di compattare l'energia nella prima componente o equivalentemente di decorrelare i dati.

Esempio: *Coppia di variabili aleatorie continue.* Consideriamo adesso due variabili aleatorie continue caratterizzate da una pdf congiunta uniforme:

$$f_{X_1X_2}(x_1, x_2) = \begin{cases} \frac{1}{\Delta_1\Delta_2} & (x_1, x_2) \in D \\ 0 & \text{altrimenti} \end{cases}$$

Le due variabili aleatorie possono cioè assumere con uguale probabilità tutti i valori che sono compresi nel dominio D mostrato in figura 16, dove assumiamo $\Delta_1 \gg \Delta_2$. Scegliamo poi una trasformazione lineare che ruoti opportunamente la densità di probabilità congiunta dei dati in ingresso in modo che i coefficienti trasformati siano tra loro incorrelati e l'energia del segnale risulti concentrata soprattutto nel primo coefficiente. Ciò che vogliamo fare è valutare la distorsione utilizzando la quantizzazione uniforme prima sui dati originali (a) e poi su quelli trasformati (b).

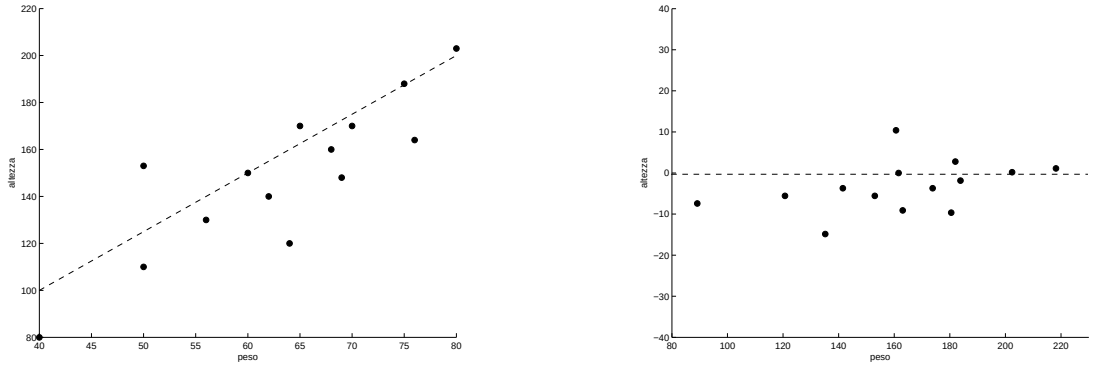


Figura 15: Rappresentazione grafica della coppia peso e altezza.

Cominciamo con l'osservare che anche le singole variabili aleatorie X_1 e X_2 hanno distribuzione uniforme, infatti ricordando che:

$$f_{X_1}(x_1) = \int_{-\infty}^{+\infty} f_{X_1X_2}(x_1, x_2)\, dx_2$$

risulta:

$$f_{X_1}(x_1) = \int_{x_1-\frac{\Delta_2}{\sqrt{2}}}^{x_1+\frac{\Delta_2}{\sqrt{2}}} \frac{1}{\Delta_1\Delta_2}\, dx_2 = \frac{\sqrt{2}}{\Delta_1} \tag{25}$$

In realtà, ciò non è rigorosamente vero ai bordi della pdf, tuttavia nell'ipotesi in cui $\Delta_1 \gg \Delta_2$ possiamo ritenere valida la (27) e dire che:

$$X_1 \sim U\left[-\frac{\Delta_1}{2\sqrt{2}}, \frac{\Delta_1}{2\sqrt{2}}\right]$$

di conseguenza la sua varianza è:

$$\sigma_1^2 \equiv E[X_1^2] = \int_{-\infty}^{+\infty} x_1^2 f_{X_1}(x_1)\, dx_1 = \int_{-\frac{\Delta_1}{2\sqrt{2}}}^{+\frac{\Delta_1}{2\sqrt{2}}} x_1^2 \frac{\sqrt{2}}{\Delta_1}\, dx_1 = \frac{\sqrt{2}}{\Delta_1}\left[\frac{x_1^3}{3}\right]_{-\frac{\Delta_1}{2\sqrt{2}}}^{\frac{\Delta_1}{2\sqrt{2}}} = \frac{\Delta_1^2}{24}$$

Analogamente possiamo affermare che $X_2 \sim U\left[-\frac{\Delta_1}{2\sqrt{2}}, \frac{\Delta_1}{2\sqrt{2}}\right]$ con $\sigma_2^2 = \frac{\Delta_1^2}{24}$. A questo punto se usiamo un quantizzatore uniforme per le due variabili aleatorie, essendo $\sigma_1^2 = \sigma_2^2 = \sigma^2$, la distorsione corrispondente è:

$$D_1(R_1) = \sigma^2\, 2^{-2R_1} \qquad D_2(R_2) = \sigma^2\, 2^{-2R_2}$$

per cui quella totale è:

$$D(R_1, R_2) = D_1(R_1) + D_2(R_2) = \sigma^2\,(2^{-2R_1} + 2^{-2R_2}) \qquad \text{con } R_1 + R_2 = R$$

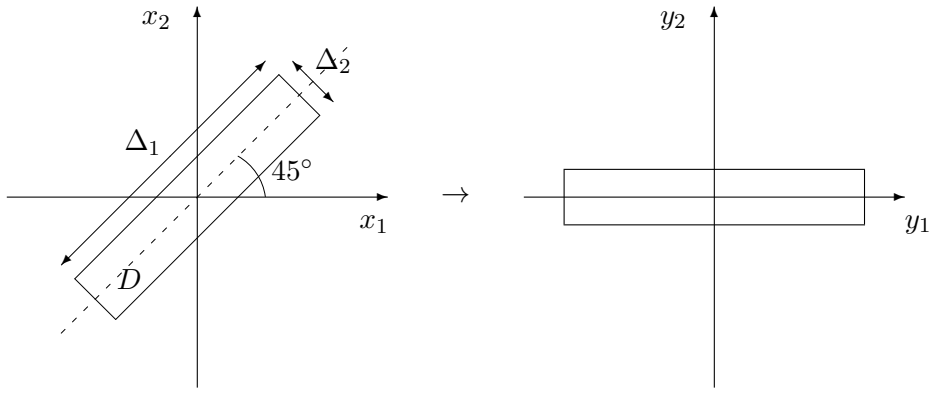


Figura 16: pdf congiunta, componenti correlate (a) componenti incorrelate (b)

Costruiamo adesso una tabella in cui mostriamo che avendo a disposizione 4 bit e assegnando le risorse uniformemente alle due variabili si raggiunge una distorsione complessiva pari a $D = \frac{\sigma^2}{8}$.

R_1	R_2	R	D_1	D_2	D
0	0	0	σ^2	σ^2	$2\sigma^2$
1	0	1	$\frac{\sigma^2}{4}$	σ^2	$\frac{5\sigma^2}{4}$
1	1	2	$\frac{\sigma^2}{4}$	$\frac{\sigma^2}{4}$	$\frac{\sigma^2}{2}$
2	1	3	$\frac{\sigma^2}{16}$	$\frac{\sigma^2}{4}$	$\frac{5\sigma^2}{16}$
2	2	4	$\frac{\sigma^2}{16}$	$\frac{\sigma^2}{16}$	$\frac{\sigma^2}{8}$

Ripetiamo questo stesso discorso operando sui dati trasformati, con una matrice di trasformazione (in questo caso $\phi = \pi/4$):

$$\mathbf{A} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix}$$

In tal caso le pdf delle due variabili aleatorie sono rigorosamente uniformi:

$$Y_1 \sim U\left[-\frac{\Delta_1}{2}, \frac{\Delta_1}{2}\right] \qquad Y_2 \sim U\left[-\frac{\Delta_2}{2}, \frac{\Delta_2}{2}\right]$$

con $\sigma_1^2 = \frac{\Delta_1^2}{12} = 2\sigma^2$ e $\sigma_2^2 = \frac{\Delta_2^2}{12} = \epsilon \ll \sigma_1^2$. Ricalcoliamo la distorsione dopo questa trasformazione, facendo però attenzione al fatto che adesso conviene assegnare le risorse esclusivamente alla prima variabile aleatoria, dato che la seconda ha già varianza trascurabile. In tal modo la distorsione complessiva si riduce a $D = \frac{\sigma^2}{128}$, come mostrato nella seguente tabella:

R_1	R_2	R	D_1	D_2	D
0	0	0	$2\sigma^2$	ϵ	$\sim 2\sigma^2$
1	0	1	$\frac{\sigma^2}{2}$	ϵ	$\simeq \frac{\sigma^2}{2}$
2	0	2	$\frac{\sigma^2}{8}$	ϵ	$\simeq \frac{\sigma^2}{8}$
3	0	3	$\frac{\sigma^2}{32}$	ϵ	$\simeq \frac{\sigma^2}{32}$
4	0	4	$\frac{\sigma^2}{128}$	ϵ	$\simeq \frac{\sigma^2}{128}$

Grazie a questo esempio si comprende che, se esiste una correlazione fra le variabili aleatorie, è senz'altro conveniente utilizzare una trasformata che le decorreli e che allo stesso compatti l'energia in una sola di esse (o nel numero minore possibile, se sono numerose). Si comprende anche che, a valle della trasformazione, bisogna stare bene attenti a come si assegnano le risorse di quantizzazione per ottenere la minima distorsione complessiva. Di seguito ci porremo dunque l'obiettivo di studiare qual è il modo ottimo per assegnare le risorse di codifica alle variabili aleatorie e come scegliere la matrice di trasformazione.

5.2 Schema di codifica basato su trasformata

Supponiamo che la sorgente emetta una sequenza di variabili aleatorie identicamente distribuite, raggruppate in blocchi di lunghezza k: $\mathbf{X} = [X_1, X_2, \ldots, X_k]$, ognuno dei quali viene sottoposto

ad una trasformazione lineare, identificata dalla matrice $\mathbf{A}$ (figura 17):

$$\mathbf{Y} = \mathbf{A}\mathbf{X} \tag{26}$$

In fase di decodifica a causa della quantizzazione non si può che recuperare una versione approssimata $\widehat{\mathbf{X}}$, sottoponendo il vettore $\widehat{\mathbf{Y}}$ dei coefficienti quantizzati alla trasformazione descritta dalla matrice inversa $\mathbf{A}^{-1}$:

$$\widehat{\mathbf{X}} = \mathbf{A}^{-1}\widehat{\mathbf{Y}} \tag{27}$$

E' utile sottolineare che, avendo considerato trasformazioni ortogonali, vale la (24) per cui l'energia dell'errore di quantizzazione è la stessa nel dominio originario ed in quello trasformato. Questa considerazione giustifica una strategia di progetto volta a minimizzare l'energia dell'errore di quantizzazione dei coefficienti trasformati, cioè: $D = E[\|\mathbf{Y} - \widehat{\mathbf{Y}}\|^2]$. Il progetto della codifica con trasformata consiste quindi in due passi fondamentali:

- individuare una trasformata (nell'ambito di quelle lineari unitarie) che permetta di compattare efficientemente l'energia.
- assegnare opportunamente le risorse, scegliere cioè i tassi di codifica con cui quantizzare le componenti del vettore trasformato in modo da minimizzare la distorsione;

Ci occuperemo innazitutto di come allocare in modo ottimo i bit disponibili per poi affrontare il problema della trasformata.

5.3 Allocazione ottima delle risorse

Per meglio comprendere come deve avvenire l'allocazione dei bit si supponga che il vettore di variabili aleatorie $X_1, X_2, \ldots, X_k$ sia a media nulla, e che ogni X_i sia caratterizzata da una diversa varianza σ_i^2. Ci poniamo l'obiettivo di quantizzare scalarmente con un tasso R_i ogni componente del vettore usando un totale di R bit di codifica. Il nostro problema quindi è quello

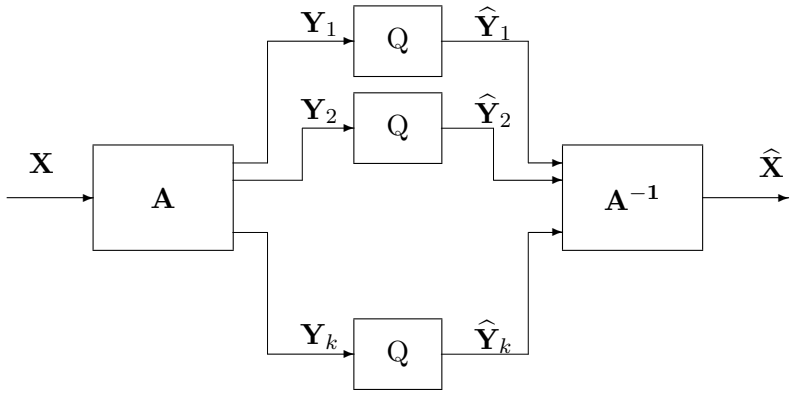


Figura 17: Schema a blocchi

di determinare i valori ottimi dei tassi di codifica $R_1, R_2, \ldots, R_k$ in modo da minimizzare la distorsione D, nell'ipotesi in cui:

$$\sum_{i=1}^{k} R_i \leq R \tag{28}$$

Osserviamo che la distorsione totale si può esprimere come:

$$\begin{aligned} D = E[\|\mathbf{X} - \widehat{\mathbf{X}}\|^2] &= E[\sum_{i=1}^{k} (X_i - \widehat{X}_i)^2] \\ &= \sum_{i=1}^{k} E[(X_i - \widehat{X}_i)^2] = \sum_{i=1}^{k} D_i \end{aligned}$$

e ricordiamo che la distorsione relativa ad ogni singola variabile aleatoria in ipotesi di quantizzazione ad elevata risoluzione è pari a:

$$D_i = h_i \, \sigma_i^2 \, 2^{-2R_i}$$

Quindi il nostro obiettivo è minimizzare la quantità:

$$D = \sum_{i=1}^{k} h_i \, \sigma_i^2 \, 2^{-2R_i}$$

con il vincolo espresso dalla (28). Questo è un problema di minimizzazione vincolata, che può essere risolto attraverso il metodo dei moltiplicatori di Lagrange. Introduciamo il funzionale:

$$J(\lambda; \mathbf{R}) = D + \lambda \left(\sum_{i=1}^{k} R_i - R \right)$$

Derivando rispetto a R_i ed uguagliando a zero si ottiene:

$$\frac{dJ}{dR_i} = \sigma_i^2 \, h_i (-2 \ln 2) 2^{-2R_i} + \lambda = 0$$

da cui si ottiene:

$$R_i = \frac{1}{2} \log_2(\sigma_i^2 \, h_i) + \lambda' \tag{29}$$

dove $\lambda' = \frac{1}{2} \log_2 \frac{2 \ln 2}{\lambda}$. Per calcolare λ' basta imporre il vincolo (28) con la condizione di uguaglianza, il che equivale ad imporre che anche la derivata di J rispetto a λ si annulli:

$$\begin{aligned} R = \sum_{i=1}^{k} R_i &= \sum_{i=1}^{k} \left[\frac{1}{2} \log_2(\sigma_i^2 \, h_i) + \lambda' \right] \\ &= \frac{1}{2} \log_2 \prod_{i=1}^{k} (\sigma_i^2 \, h_i) + k\lambda' \end{aligned}$$

Da cui si ricava che

$$\begin{aligned}\lambda' &= \frac{R}{k} - \frac{1}{2}\log_2 \prod_{i=1}^{k} (\sigma_i^2\, h_i)^{1/k} \\ &= \overline{R} - \frac{1}{2}\log_2 \sigma_{\mathrm{GM}}^2 h_{\mathrm{GM}} \end{aligned} \tag{30}$$

dove $\overline{R} = R/k$ è il numero medio di bit per componente, mentre h_{GM} e σ_{GM}^2 sono, rispettivamente, la media geometrica dei fattori di forma e delle varianze

$$h_{\mathrm{GM}} = \left[\prod_{i=1}^{k} h_i\right]^{1/k} \qquad \sigma_{\mathrm{GM}}^2 = \left[\prod_{i=1}^{k} \sigma_i^2\right]^{1/k}$$

sostituendo la (30) nella (29) si ottiene la formula di Huang e Schultheiss per l'allocazione ottima delle risorse:

$$R_i = \frac{R}{k} + \frac{1}{2}\log_2 \frac{\sigma_i^2\, h_i}{\sigma_{\mathrm{GM}}^2\, h_{\mathrm{GM}}} \tag{31}$$

Questa relazione ci dice che il numero di bit da assegnare alla i-esima componente è dato dal numero medio di bit per componente $\overline{R}$, corrispondente ad un'allocazione equa delle risorse tra tutte le componenti, più un termine correttivo, che come già messo in luce, dipende dalla varianza della componente (supponiamo qui per semplicità di analisi che i fattori di forma siano tutti uguali) ed è positivo o negativo. In particolare è positivo se X_i è in un certo senso meno "predicibile della media" cioè se $\sigma_i^2 > \sigma_{\mathrm{GM}}^2$. In caso contrario il termine correttivo è negativo, e X_i si vedrà assegnate meno risorse della media. E' interessante notare che, in caso di allocazione ottima delle risorse le distorsioni sono tutte uguali, infatti risulta:

$$\begin{aligned} D_i = \sigma_i^2 h_i 2^{-2R_i} &= \sigma_i^2 h_i 2^{-2\left[\frac{R}{k} + \frac{1}{2}\log_2 \frac{\sigma_i^2 h_i}{\sigma_{\mathrm{GM}}^2 h_{\mathrm{GM}}}\right]} \\ &= \sigma_i^2 h_i 2^{-2\frac{R}{k}} \cdot \frac{\sigma_{\mathrm{GM}}^2 h_{\mathrm{GM}}}{\sigma_i^2 h_i} \\ &= \sigma_{\mathrm{GM}}^2\, h_{\mathrm{GM}} 2^{-2\overline{R}} \end{aligned}$$

La distorsione, quindi, è identica per tutte le componenti, e le prestazioni sono tanto migliori quanto più piccola è la media geometrica delle varianze. La relativa distorsione totale risulta essere:

$$D_{\min} = k\, \sigma_{\mathrm{GM}}^2\, h_{\mathrm{GM}} 2^{-2\overline{R}}$$

Vogliamo ora confrontare tale distorsione con quella che avremmo se invece di assegnare le risorse nel modo ottimo, avessimo assegnato ad ogni componente lo stesso tasso $\overline{R}$. Si trova:

$$D_{\mathrm{tot}} = \sum_{i=1}^{k} h_i \sigma_i^2 2^{-2\overline{R}}$$

Consideriamo adesso il rapporto tra le due distorsioni:

$$\frac{D_{\text{tot}}}{D_{\min}} = \frac{\sum_{i=1}^{k} h_i \sigma_i^2 2^{-2\overline{R}}}{k\,\sigma_{GM}^2 h_{GM} 2^{-2\overline{R}}} = \frac{\frac{1}{k}\sum_{i=1}^{k} h_i \sigma_i^2 2^{-2\overline{R}}}{\sigma_{GM}^2 h_{GM} 2^{-2\overline{R}}}$$

Al numeratore è presente la media aritmetica delle quantità: $h_i\sigma_i^2$, mentre al denominatore c'è la loro media geometrica. Ricordiamo che la media aritmetica di un insieme di numeri non negativi è sempre maggiore della sua media geometrica, per cui questo rapporto, indicato anche come *guadagno di codifica*, CG (coding gain) è sempre maggiore o uguale ad 1. La (31) ci dice dunque che la tecnica di allocazione delle risorse è tanto più efficace quanto più la media aritmetica di $h_i\sigma_i^2$ è maggiore della sua media geometrica. A parità di media aritmetica, questo avviene quanto più i valori di $h_i\sigma_i^2$ sono distribuiti disomogeneamente. In altre parole, mentre non c'è (ovviamente) nessun vantaggio ad applicare la (31) ad un vettore di variabili aleatorie identicamente distribuite, le prestazioni RD del sistema di compressione migliorano se le componenti del vettore aleatorio sono distribuite in modo disomogeneo, il che avviene ad esempio se l'energia del vettore è concentrata in poche componenti. Con riferimento alle tecniche di codifica con trasformata, è quindi desiderabile che esse soddisfino il più possibile questo obiettivo.

Osserviamo che la presenza di un termine additivo potenzialmente negativo, significa che la formula (41) potrebbe produrre come risultato dei valori negativi. Se si ottiene qualche R_i negativo questo si deve porre uguale a zero e ripetere il procedimento. Un'altra precisazione da fare è che in generale i valori di R_i che si ottengono non sono numeri interi, in tal caso R_i è possibile approssimare all'intero più vicino in modo comunque da rispettare il vincolo $\sum_{i=1}^{k} R_i = R$, si ottiene però così un sub–ottimo.

5.4 Trasformata di Karhunen-Loéve

La trasformazione ottima, nel senso che fornisce coefficienti incorrelati e che realizza la massima compattazione di energia, è quella di Karhunen-Loéve (KLT), nota anche in letteratura con il nome di Analisi delle Componenti Principali (PCA). Consideriamo il vettore aleatorio $\mathbf{X} = (X_1, X_2, \ldots, X_k)^T$ (vettore colonna), che senza perdita di generalità, possiamo supporre a media nulla. Sia $\mathbf{R}_X$ la sua matrice di correlazione:

$$\mathbf{R}_X = E[\mathbf{X}\cdot\mathbf{X}^T] = E\left[\begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_k \end{pmatrix}\begin{pmatrix} X_1, X_2, \ldots, X_k \end{pmatrix}\right]$$

$$= E\begin{bmatrix} X_1^2 & X_1X_2 & \ldots & X_1X_k \\ X_2X_1 & X_2^2 & \ldots & X_2X_k \\ \vdots & \vdots & \ldots & \vdots \\ X_kX_1 & X_kX_2^2 & \ldots & X_k^2 \end{bmatrix} = \begin{bmatrix} \overline{X_1^2} & \overline{X_1X_2} & \ldots & \overline{X_1X_k} \\ \overline{X_2X_1} & \overline{X_2^2} & \ldots & \overline{X_2X_k} \\ \vdots & \vdots & \ldots & \vdots \\ \overline{X_kX_1} & \overline{X_kX_2^2} & \ldots & \overline{X_k^2} \end{bmatrix}$$

Tale matrice gode delle seguenti proprietà:

- è simmetrica;
- è semidefinita positiva, cioè $\mathbf{a}^T\mathbf{R}_X\mathbf{a} \geq 0$ comunque si scelga un vettore $\mathbf{a}$ k-dimensionale;

Quest'ultima proprietà ci garantisce che gli autovalori di $\mathbf{R}_X$ sono reali e non negativi. Si può poi dimostrare che se nessuna delle componenti di $\mathbf{X}$ si può ottenere come combinazione linerare di altre, allora la matrice è *definita* positiva, per cui tutti gli autovalori sono strettamente positivi. D'ora in poi assumeremo che questa ipotesi sia verificata, e indicheremo con $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_k > 0$ gli autovalori di $\mathbf{R}_X$, con l'implicita assunzione di averli ordinati in senso decrescente. Inoltre, essendo la matrice simmetrica e reale di ordine k, essa è dotata di k autovettori[3] distinti e ortonormali $\{\mathbf{u}_i\}_{i=1}^k$, pertanto la matrice:

$$\mathbf{U} = [\mathbf{u}_1 \,|\, \mathbf{u}_2 \,|\, \ldots \,|\, \mathbf{u}_k]$$

che ha per colonne proprio tali autovettori risulta ortonormale $\mathbf{U}^{-1} = \mathbf{U}^T$. La matrice $\mathbf{R}_X$ ammette allora la seguente decomposizione:

$$\mathbf{R}_X = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \tag{32}$$

dove $\mathbf{\Lambda} = \text{diag}\,\{\lambda_1, \lambda_2, \ldots, \lambda_k\}$. La trasformata di Karhunen-Loéve si definisce scegliendo come matrice di trasformazione la matrice $\mathbf{U}^T$, le cui *righe* sono gli autovettori normalizzati di $\mathbf{R}_X$, quindi:

$$\mathbf{Z} = \mathbf{U}^T\mathbf{X} \tag{33}$$

ed è chiaramente una trasformata ortonormale. Le principali proprietà della KLT sono:

1. è una trasformata decorrelante;
2. compatta l'energia nel modo migliore possibile.

Dimostriamo la prima proprietà. Sia $\mathbf{Z} = \mathbf{U}^T\mathbf{X}$ il vettore risultante dalla trasformazione di $\mathbf{X}$. Risulta:

$$\begin{aligned}\mathbf{R}_Z = E[\mathbf{Z}\,\mathbf{Z}^T] &= E[\mathbf{U}^T\,\mathbf{X}\,(\mathbf{U}^T\mathbf{X})^T] \\ &= E[\mathbf{U}^T\,\mathbf{X}\,\mathbf{X}^T\,\mathbf{U}] = \mathbf{U}^T E[\mathbf{X}\mathbf{X}^T]\,\mathbf{U} = \mathbf{U}^T\mathbf{R}_X\mathbf{U}\end{aligned} \tag{34}$$

d'altra parte è valida la (32) pertanto:

$$\mathbf{R}_Z = \mathbf{U}^T\mathbf{U}\mathbf{\Lambda}\mathbf{U}^T\mathbf{U} = \mathbf{U}^{-1}\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}\mathbf{U} = \mathbf{\Lambda} \tag{35}$$

La matrice di correlazione di $\mathbf{Z}$ è quindi diagonale, le correlazioni tra le variabili aleatorie che costituiscono il vettore sono tutte nulle, eccetto sulla diagonale principale in cui sono presenti le varianze che coincidono proprio con gli autovalori.

[3] Ricordiamo che, data una matrice $\mathbf{Q}$, si dice che $\mathbf{u}_i$ è un suo autovettore se per qualche $\lambda_i \in \mathbb{R}$ risulta

$$\mathbf{Q}\,\mathbf{u}_i = \lambda_i\mathbf{u}_i \qquad \text{ovvero} \qquad [\mathbf{Q} - \lambda_i\mathbf{I}]\,\mathbf{u}_i = 0,$$

in tal caso λ_i è detto l'autovalore corrispondente all'autovettore $\mathbf{u}_i$.

Nonostante la KLT abbia le proprietà positive prima elencate, essa è poco usata nella compressione di immagini naturali. Il problema principale è che la matrice di trasformazione dipende dalle statistiche dei dati (*data dependent*), e quindi, a meno che esse non siano note a priori (cosa che raramente accade), tale matrice deve essere determinata di volta in volta causando un aumento della complessità computazionale.

In realtà, il problema principale risiede nella scarsa affidabilità della stima della matrice di autocorrelazione dei dati. Le medie statistiche infatti vengono stimate tramite medie aritmetiche, ma tale operazione è sensata solo se il segnale in ingresso è stazionario, ipotesi non ragionevole per le immagini naturali. Infine, bisogna tener presente che la matrice di autocorrelazione non è nota al decodificatore, al quale va inviata (*side information*) assieme alla sequenza codificata, causando un incremento di dati da trasmettere. Ciò significa che l'intero paradigma della KLT mal si adatta alle immagini naturali, ma trova applicazione nella compressione di particolari immagini da telerilevamento, dette immagini multispettrali.

5.5 Trasformata coseno discreta

A causa dei problemi descritti nel paragrafo precedente la KLT viene raramente utilizzata e si preferisce, al suo posto, far uso di trasformazioni sub-ottime, come la trasformata coseno discreta (DCT). Proprietà interessanti di tale trasformata sono una bassa complessità computazionale, dovuta alle particolari caratteristiche di simmetria della matrice e alla possibilità di utilizzare algoritmi veloci FFT (Fast Fourier Transform), e una buona capacità di compattazione dell'energia; si può dimostrare, infatti, che i suoi vettori base sono molto simili a quelli della KLT per sorgenti caratterizzate da elevata correlazione.

In questa sezione definiremo la trasformata coseno discreta monodimensionale e vedremo qual è il suo legame con la trasformata discreta di Fourier (DFT). Essendo una trasformata lineare, detto $\mathbf{x}$ il vettore N-dimensionale da elaborare, l'uscita si ottiene dal prodotto matriciale:

$$\mathbf{y} = \mathbf{C}\mathbf{x}$$

ovvero:

$$\begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(N-1) \end{bmatrix} = \begin{bmatrix} c_{0,0} & c_{0,1} & \dots & c_{0,N-1} \\ c_{1,0} & c_{1,1} & \dots & c_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{N-1,0} & c_{N-1,1} & \dots & c_{N-1,N-1} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix}$$

dove $\mathbf{C}$, ha dimensioni $N \times N$, è ortonormale ed ha come generico elemento:

$$c_{k,n} = \begin{cases} \sqrt{\frac{1}{N}} & k = 0, \qquad n = 0, 1, \dots, N-1 \\ \sqrt{\frac{2}{N}} \cos[\frac{\pi k}{2N}(2n+1)] & k = 1, 2, \dots, N-1, \qquad n = 0, 1, \dots, N-1 \end{cases}$$

Notiamo come la prima riga sia costituita da un vettore costante, mentre dalla seconda riga in poi abbiamo l'espressione di sinuosidi con frequenza legata all'indice k. Questo significa che all'aumentare di k le funzioni base con cui realizzare la trasformazione sono costituite da sinusoidi che variano sempre più rapidamente. In particolare, ogni componente del vettore trasformato che può essere espressa come:

$$y(k) = \sum_{n=0}^{N-1} c_{k,n}\, x(n) \qquad k = 0, 1, \dots, N-1$$

è ottenuta proiettando il vettore $\mathbf{x}$ su ognuna di queste funzioni con frequenze $\frac{k}{2N}$ per $k = 0, 1, \ldots, N-1$, tutte multiple di una frequenza fondamentale (così come accade per la trasformata di Fourier). Chiaramente $y(0)$ rappresenterà proprio la componente continua (coefficiente DC) del vettore trasformato $\mathbf{y}$, mentre gli altri elementi $y(1), y(2), \ldots, y(N-1)$ non sono altro che le sue componenti in frequenza (coefficienti AC).

Vediamo allora come è legata la DCT alla trasformata discreta di Fourier (DFT) e per quale motivo non si usa direttamente il dominio di Fourier per la trasformazione. Innanzitutto, teniamo presente che la trasformata di Fourier restituisce sia coefficienti reali che immaginari, inoltre per come è definita si introducono inevitabilmente delle componenti frequenziali che non fanno parte del contenuto informativo del segnale. Per comprendere questo concetto, facciamo un esempio. Innanzitutto ricordiamo l'equazione di analisi per la DFT:

$$X(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi nk}{N}} \quad k = 0, 1, \ldots, N-1$$

e supponiamo di voler relizzare la 8-DFT del vettore mostrato di seguito

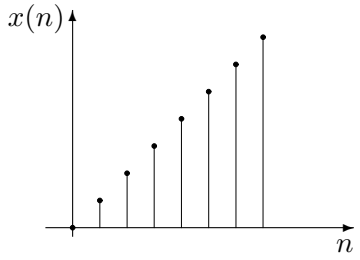


Figura 18: Segnale da trasformare.

Questo significa di fatto replicare la sequenza su tutto l'asse temporale, farne la trasformata e poi considerare solo i primi 8 coefficienti. La replica di $x(n)$ introduce nel segnale delle discontinuità lungo l'asse temporale così come si mostra in figura 20. Tali variazioni non esistono nella sequenza originaria che presenta invece variazioni lente introducendo componenti frequenziali indesiderate nel dominio trasformato e che andrebbero poi codificate. Un modo per superare questo problema è quello di costruire, non una replica periodica, bensì simmetrica, così come si mostra in figura 21.
Consideriamo allora una sola replica del segnale esteso simmetricamente e indichiamola con $y(n)$:

$$y(n) = \begin{cases} x(n) & n = 0, 1, \ldots, N-1 \\ x(2N-1-n) & n = N, \ldots, 2N-1 \end{cases} \tag{36}$$

quindi calcoliamo la DFT su $2N$ campioni, si ottiene:

$$Y(k) = \frac{1}{\sqrt{2N}} \sum_{n=0}^{2N-1} y(n)\, e^{-j\frac{2\pi\left(n+\frac{1}{2}\right)k}{2N}} \qquad k = 0, 1, \ldots, 2N-1$$

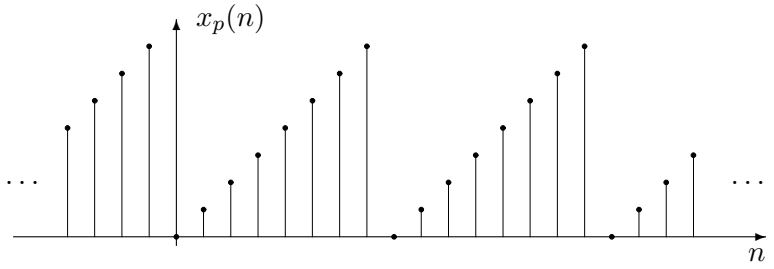


Figura 19: Estensione periodica.

Osserviamo però che l'esponente è stato modificato aggiungendo $\frac{1}{2}$. Ciò è dovuto al fatto che la replicazione della sequenza $y(n)$, cioè $x_s(n)$ non è pari rispetto all'origine. Aggiungendo $\frac{1}{2}$ si rende pari con il vantaggio di ottenere coefficienti di Fourier reali. Suddividiamo adesso la sommatoria in due quantità:

$$Y(k) = \frac{1}{\sqrt{2N}} \left[\sum_{n=0}^{N-1} y(n)\, e^{-j\frac{2\pi\left(n+\frac{1}{2}\right)k}{2N}} + \sum_{n=N}^{2N-1} y(n)\, e^{-j\frac{2\pi\left(n+\frac{1}{2}\right)k}{2N}} \right]$$

ricordando che è valida la (38) si ha:

$$Y(k) = \frac{1}{\sqrt{2N}} \left[\sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi\left(n+\frac{1}{2}\right)k}{2N}} + \sum_{n=N}^{2N-1} x(2N-1-n)\, e^{-j\frac{2\pi\left(n+\frac{1}{2}\right)k}{2N}} \right]$$

A questo punto facciamo un cambio di variabili ponendo $m = 2N - 1 - n$:

$$= \frac{1}{\sqrt{2N}} \left[\sum_{n=0}^{N-1} x(n)\, e^{-j\frac{2\pi\left(n+\frac{1}{2}\right)k}{2N}} + \sum_{m=0}^{N-1} x(m)\, e^{+j\frac{2\pi\left(m+\frac{1}{2}\right)k}{2N}} \right]$$

$$= \frac{1}{\sqrt{2N}} \sum_{n=0}^{N-1} x(n) \left[e^{-j\frac{2\pi\left(n+\frac{1}{2}\right)k}{2N}} + e^{+j\frac{2\pi\left(n+\frac{1}{2}\right)k}{2N}} \right]$$

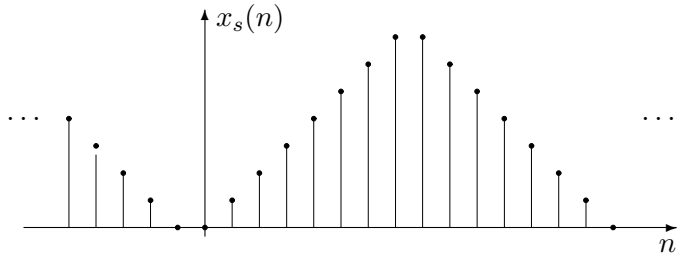


Figura 20: Estensione simmetrica.

Applicando infine la formula di Eulero si ha:

$$Y(k) = \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} x(n) \cos\left[\frac{\pi(2n+1)k}{2N}\right]$$

che corrisponde proprio alla definizione della DCT per $k \neq 0$. Per $k = 0$ il vettore è costante e deve essere normalizzato per cui si introduce il fattore $\sqrt{\frac{1}{N}}$.

5.6 Le trasformate bidimensionali

Per realizzare la trasformata di un'immagine è necessario estendere al caso bidimensionale le definizioni (21) e (22). Sia allora $x(m,n)$ il campione in posizione (m,n) di un'immagine $M \times N$, allora la trasformata lineare è definita come:

$$y(k,l) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} a_{k,l,m,n}\, x(m,n), \qquad k = 0, \cdots, M-1, \qquad l = 0, \cdots, N-1 \tag{37}$$

mentre la relazione di sintesi che ne rende possibile la ricostruzione è:

$$x(m,n) = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} b_{m,n,k,l}\, y(k,l), \qquad k = 0, \cdots, M-1, \qquad l = 0, \cdots, N-1 \tag{38}$$

La maggior parte delle trasformata lineari che si usano nella pratica sono separabili, vale a dire godono della proprietà per cui $a_{k,l,m,n} = a_{k,m} a_{l,n}$, questo significa che è possibile realizzare prima la trasformata (monodimensionale) lungo le righe e poi quella lungo le colonne (o viceversa). In tal caso la (37) diventa:

$$y(k,l) = \sum_{n=0}^{N-1} \left[\sum_{m=0}^{M-1} a_{k,m}\, x(m,n) \right] a_{l,n} \qquad k = 0, \cdots, M-1, \qquad l = 0, \cdots, N-1 \tag{39}$$

mentre la (38):

$$x(m,n) = \sum_{l=0}^{N-1} \left[\sum_{k=0}^{M-1} b_{k,m}\, y(k,l) \right] b_{l,n} \qquad k = 0, \cdots, M-1, \qquad l = 0, \cdots, N-1 \tag{40}$$

e in forma matriciale risulta:

$$\mathbf{y} = \mathbf{A}_M \mathbf{x} \mathbf{A}_N^T \tag{41}$$

dove $\mathbf{A}_M$ è una matrice $M \times M$ e $\mathbf{A}_N$ è una matrice $N \times N$. Equivalentemente in fase di sintesi si ha:

$$\mathbf{x} = \mathbf{B}_M \mathbf{y} \mathbf{B}_N^T \tag{42}$$

Essendo la DCT una trasformata separabile, l'estensione al caso bidimensionale risulta molto semplice in quanto basta applicare la (41) per la trasformata diretta, ponendo $\mathbf{A} = \mathbf{C}$, e la (42) per quella inversa con $\mathbf{B} = \mathbf{C}^T$, essendo $\mathbf{C}$ una matrice unitaria. In figura 21 si mostra un'immagine su scala di grigi, di cui è stata calcolata la trasformata coseno discreta, rappresentata in falsi colori per una migliore visualizzazione. Notiamo come la DCT concentri l'energia relativa alle componenti in bassa frequenza nella regione in alto a sinistra dell'immagine.

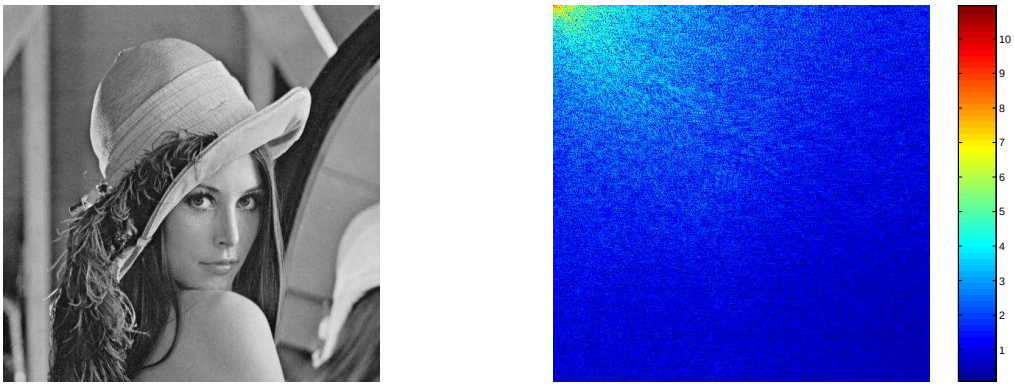


Figura 21: Immagine originale e DCT bidimensionale.

6 Lo standard JPEG

La sigla JPEG identifica una commissione di esperti denominata Joint Photographic Expert Group, formata nel 1986 con lo scopo di stabilire uno standard di compressione per le immagini a tono continuo (cioè di tipo fotografico) sia a colori che in bianco e nero. Il lavoro di questa commissione ha portato alla definizione di una complessa serie di algoritmi, approvata come standard ISO nell'agosto del 1990 e successivamente divenuta la raccomandazione T.81 (9/92) dell'ITU, International Telecommunication Union, organizzazione che promuove gli standard per le telecomunicazioni, e componente del CCITT precedentemente nominata. JPEG è dunque uno standard industriale e non va confuso con il formato di file JPG, che rappresenta di volta in volta, a seconda del software che lo implementa, un sottoinsieme variabile e non sempre universalmente compatibile dello standard di riferimento. Tale standard si pone infatti come obiettivo la definizione di una tecnica di compressione adattabile.

L'algoritmo prevede la possibilità di operare sia in modalità lossy (con perdite) che lossless (senza perdite). La compressione lossy può essere effettuata mediante le seguenti modalità:

1. *Sequenziale.* E' la modalità base (Baseline JPEG), nella quale l'immagine è compressa, con perdita di qualità in un solo passaggio operando per righe e procedendo dall'alto verso il basso (figura 1).

2. *Progressiva.* La modalità sequenziale presenta l'inconveniente di elaborare, trasmettere e ricostruire l'immagine una riga per volta, fornendo in ricezione solo una visione parziale fino al completo trasferimento dei dati. Per superare questo limite JPEG prevede che l'immagine possa essere codificata attraverso passaggi successivi ciascuno dei quali aumenta la qualità dell'immagine ricostruita. E' una tecnica adatta a quelle applicazioni nelle quali si preferisce avere subito un'immagine di bassa qualità, che subisce progressivi miglioramenti man mano che i dati vengono decodificati (figura 2).

3. *Gerarchica.* Con questa modalità l'immagine viene elaborata con risoluzioni spaziali via via crescenti (per esempio 256×256, 512×512 e 1024×1024) in modo che sia possibile ricostruirla alla risoluzione più bassa, senza che sia necessario decomprimere l'immagine alla massima risoluzione possibile. E' anche possibile prevedere una codifica progressiva per le immagini a diversa risoluzione.

Figura 22: Codifica sequenziale.

Figura 23: Codifica progressiva.

Nonostante le specifiche JPEG includano tecniche differenti di compressione, il metodo di codifica più diffuso è quello sequenziale. Di seguito si effettuerà una descrizione dettagliata di questo metodo e delle ragioni che hanno portato a determinate scelte nella definizione dello standard. Per una più efficace analisi dividiamo l'algoritmo di compressione nei seguenti passi che, poi, andremo a trattare separatamente:

- trasformata coseno discreta;
- quantizzazione;
- codifica entropica.

6.1 DCT a blocchi

JPEG usa la trasformata coseno discreta per compattare l'energia in un numero piccolo di coefficienti e realizzare quindi una più efficiente quantizzazione scalare. In realtà, la trasformata è applicata a blocchi disgiunti piuttosto che su tutta l'immagine e il motivo risiede nella natura delle funzioni usate nella trasformazione che, essendo di tipo sinusoidale, si prestano a rappresentare bene solo i segnali che variano lentamente. Infatti, maggiori sono le dimensioni dei blocchi più aumenta la probabilità che siano presenti brusche discontinuità che, nel dominio trasformato, vengono disperse su un elevato numero di coefficienti di valore prossimo a zero. La successiva quantizzazione potrebbe annullare tali coefficienti, e quindi nell'immagine ricostruita tali regioni risulterebbero fortemente distorte. E' necessario quindi operare su blocchi piccoli, ma non troppo, per garantire comunque un'elevata efficienza di codifica. Il compromesso tra queste diverse esigenze ha portato a scegliere di partizionare l'immagine in blocchi di dimensioni

8×8. E' bene notare che le dimensioni dell'immagine potrebbero non essere multiple di 8, in tal caso l'algoritmo prevede di modificare le dimensioni replicando l'ultima colonna a destra dell'immagine e l'ultima riga in basso fino a che le dimensioni lungo le due direzioni risultino multiple di 8. Questa operazione fa in modo che la luminosità media del blocco 8×8 non vari significativamente, rispetto al caso in cui si effettui un riempimento con zeri.

Prima di applicare la DCT a blocchi, i dati vengono resi a media nulla sottraendo all'immagine il valore 2^{P-1}, dove P è il numero di bit usato per rappresentare ogni pixel. Se, per esempio, i dati appartenevano al range $[0, \ldots, 255]$ dopo la sottrazione apparterranno ad un intervallo quasi simmetrico rispetto all'origine $[-128, \ldots, 127]$.

A questo punto, definiamo la trasformata coseno discreta modimensionale (DCT-1D) per poi estendere i concetti al caso bidimensionale (DCT-2D). La DCT-1D di un vettore $\mathbf{x} = (x_0, x_1, \cdots, x_7)$ lungo 8 fornisce il vettore:

$$\mathbf{y} = \mathbf{C}_8 \, \mathbf{x}$$

ancora lungo 8, o più esplicitamente le componenti

$$\begin{aligned} y_0 &= c_{0,0}x_0 + c_{0,1}x_1 + \cdots + c_{0,7}x_7 \\ y_1 &= c_{1,0}x_0 + c_{1,1}x_1 + \cdots + c_{1,7}x_7 \\ \ldots &= \ldots \\ y_7 &= c_{7,0}x_0 + c_{7,1}x_1 + \cdots + c_{7,7}x_7 \end{aligned}$$

il generico elemento $c_{k,n}$ individua la matrice unitaria $\mathbf{C}_8$ ed è così definito:

$$c_{k,n} = \begin{cases} \frac{1}{2\sqrt{2}} & k = 0, \qquad n = 0, 1, \ldots, 7 \\ \frac{1}{2} \cos[\frac{\pi k(2n+1)}{16}] & k = 1, 2, \ldots, 7, \qquad n = 0, 1, \ldots, 7 \end{cases} \tag{43}$$

Il vettore trasformato $\mathbf{y}$ si può interpretare in termini di espansione rispetto alle righe della matrice di trasformazione, che rappresentano i vettori della base dello spazio 8-dimensionale rispetto cui si valutano le nuove coordinate. Pertanto ogni elemento del vettore non è altro che la proiezione del segnale su una componente della base, ovvero il prodotto scalare tra i due elementi; per esempio il primo coefficiente del segnale trasformato è

$$y_0 = \sum_n c_{0,n} x_n = < \mathbf{c_0}, \mathbf{x} >$$

e risulta essere tanto più elevato quanto più il segnale in ingresso è simile ad un vettore costante. Si può notare come le righe della matrice abbiano un andamento sinusoidale con frequenza crescente all'aumentare di k, di conseguenza un valore elevato del coefficiente trasformato per $k = 7$ indica che il segnale in ingresso presentava un andamento molto simile ad una sinusoide velocemente variabile (Figura 24).

L'estensione al caso bidimensionale è piuttosto semplice, infatti, grazie alla proprietà di separabilità della trasformata di Fourier bidimensionale, anche la DCT-2D è separabile e può essere calcolata operando prima lungo le righe e poi lungo le colonne di ogni blocco, usando la seguente relazione:

$$\mathbf{Y} = \mathbf{C}_8 \, \mathbf{X} \mathbf{C}_8^T \tag{44}$$

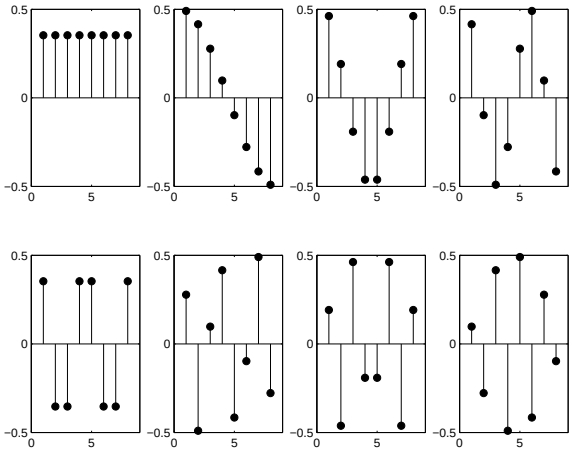


Figura 24: vettori riga della matrice $\mathbf{C}$ per $N = 8$.

dove $\mathbf{X}$ è il blocco 8×8 in ingresso e $\mathbf{Y}$ è quello che si ottiene in uscita. E' molto comodo interpretare anche la trasformata bidimensionale come espansione rispetto ad opportune matrici, in tal modo ogni pixel appartenente al blocco 8×8 in uscita si può vedere come la proiezione del blocco in ingresso su una matrice 8×8 opportunamente definita. Infatti riscrivendo la (44) in termini matriciali risulta:

$$\begin{bmatrix} y_{0,0} & y_{0,1} & \cdots & y_{0,7} \\ y_{1,0} & y_{1,1} & \cdots & y_{1,7} \\ \vdots & \vdots & \ddots & \vdots \\ y_{7,0} & y_{1,1} & \cdots & y_{7,7} \end{bmatrix} =$$

$$\begin{bmatrix} c_{0,0} & c_{0,1} & \cdots & c_{0,7} \\ c_{1,0} & c_{1,1} & \cdots & c_{1,7} \\ \vdots & \vdots & \ddots & \vdots \\ c_{7,0} & c_{7,1} & \cdots & c_{7,7} \end{bmatrix} \begin{bmatrix} x_{0,0} & x_{0,1} & \cdots & x_{0,7} \\ x_{1,0} & x_{1,1} & \cdots & x_{1,7} \\ \vdots & \vdots & \ddots & \vdots \\ x_{7,0} & x_{7,1} & \cdots & x_{7,7} \end{bmatrix} \begin{bmatrix} c_{0,0} & c_{1,0} & \cdots & c_{7,0} \\ c_{0,1} & c_{1,1} & \cdots & c_{7,1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{0,7} & c_{1,7} & \cdots & c_{7,7} \end{bmatrix}$$

Proviamo adesso a calcolare esplicitamente il valore $y_{0,0}$:

$$y_{0,0} = c_{0,0}^2 x_{0,0} + \cdots + c_{0,7} c_{0,0} x_{7,0} + c_{0,0} c_{0,1} x_{0,1} + \cdots + c_{0,7} c_{0,1} x_{7,1} + \cdots + c_{0,0} c_{0,7} x_{0,7} + \cdots + c_{0,7}^2 x_{7,7}$$

Se si definisce

$$\mathbf{A}_{0,0} = \begin{bmatrix} c_{0,0}^2 & c_{0,1} c_{0,0} & \cdots & c_{0,7} c_{0,0} \\ c_{0,0} c_{0,1} & c_{0,1}^2 & \cdots & c_{0,7} c_{0,1} \\ \vdots & \vdots & \ddots & \vdots \\ c_{0,0} c_{0,7} & c_{0,1} c_{0,7} & \cdots & c_{0,7}^2 \end{bmatrix}$$

si può affermare che il valore $y_{0,0}$ si ottiene moltiplicando punto-punto la matrice $\mathbf{A}_{0,0}$ con il

blocco di dati $\mathbf{X}$ e poi sommando il risultato. D'altra parte è valida la (43), per cui

$$\mathbf{A}_{0,0} = \frac{1}{8}\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix}$$

In conclusione, possiamo esprimere $y_{0,0}$, indicata anche come componente continua (DC) (e a questo punto se ne comprende il motivo) come:

$$y_{0,0} = \frac{1}{8}\sum_{i=0}^{7}\sum_{j=0}^{7} x(i,j) \tag{45}$$

La matrice $\mathbf{A}_{0,0}$, rappresentata come immagine nel primo blocco in alto a sinistra di figura 25, è costituita da elementi costanti, e quindi fornirà un valore elevato se il blocco dell'immagine varia molto lentamente, si noti poi che questa matrice può anche essere determinata come il prodotto matriciale tra i due seguenti vettori:

$$\mathbf{A}_{0,0} = \begin{bmatrix} c_{0,0} \\ c_{0,1} \\ \vdots \\ c_{0,7} \end{bmatrix} [c_{0,0}, c_{0,1} \cdots c_{0,7}] = \mathbf{c}_0\mathbf{c}_0^T$$

Chiaramente questo discorso può essere esteso al calcolo di tutti gli altri coefficienti del blocco trasformato (detti anche AC), che possono essere interpretati come la proiezione della porzione di immagine di dimensioni 8×8 su funzioni base bidimensionali $\mathbf{A}_{i,j}$ (figura 25) che variano sempre più velocemente per $i = 1, \cdots, 8$ (lungo le righe) e $j = 1, \cdots, 8$ (lungo le colonne). Se volessimo visualizzare una riga o una colonna della matrice mostrata in figura 25 otterremmo segnali sinusoidali che variano via via più velocemente lungo le due direzioni. Di conseguenza, se il blocco da trasformare è costituito, per esempio, da lente variazioni lungo la direzione orizzontale,

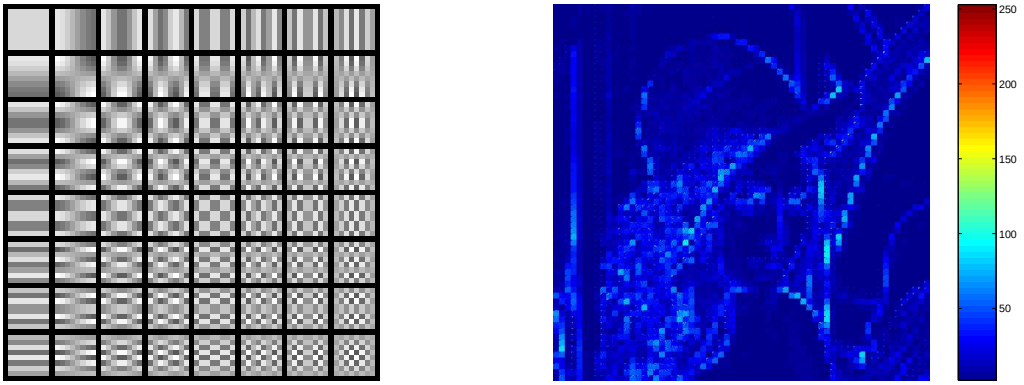
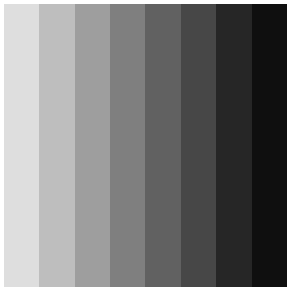


Figura 25: Matrici bidimensionali $\mathbf{A}_{i,j}$ su cui vengono proiettati i blocchi 8×8 dell'immagine e relativa trasformata.



$$\begin{bmatrix} -88 & 550.3 & 11 & 43.3 & -6 & 13.8 & 21.4 & 10.1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figura 26: Blocco 8×8 e relativi coefficienti DCT.

otterremo elevati coefficienti proprio lungo tale direzione, laddove il blocco si confronterà con blocchi più simili (figura 26).

Infine, ricordiamo che la DCT-2D è perfettamente reversibile, per cui l'immagine originale si può ottenere operando su ogni blocco mediante la seguente trasformazione:

$$\mathbf{X} = \mathbf{C}_8^T \mathbf{Y} \mathbf{C}_8 \tag{46}$$

In figura 27 si mostra l'immagine originale e quella ricostruita conservando solo la componente continua di ogni blocco. Il fatto che essa rappresenti una versione più grossolana dell'immagine conferma il fatto che le immagini comuni hanno il loro contenuto energetico concentrato alle basse frequenze e che la trasformazione DCT rappresenti un buon strumento per comprimerle.

6.2 Quantizzazione

Lo standard JPEG utilizza un quantizzatore uniforme che ha tra i livelli di restituzione anche lo zero (*Midtread Quantizer*). Il passo di quantizzazione si differenzia a seconda della posizione dell'elemento nel blocco 8×8, ma è fisso qualunque sia l'immagine da comprimere. I passi di quantizzazione per i 64 coefficienti del blocchetto sono organizzati in una tabella in cui è definita la matrice di quantizzazione, mostrata in figura 28. Questo tipo di quantizzazione, usato anche

Figura 27: Immagine originale e immagine ricostruita conservando solo la componente continua di ogni blocco.

nello standard di compressione MPEG, cerca di ridurre la ridondanza psicofisica basandosi sul comportamento dell'apparato visivo umano. L'occhio umano, infatti, è più sensibile alle basse frequenze spaziali piuttosto che a quelle elevate, quindi, una distorsione introdotta in fase di codifica sulle basse frequenze è apprezzata dall'occhio molto di più che non un'identica distorsione sulle alte frequenze. Per questo motivo lo standard JPEG adotta una quantizzazione differente a seconda dell'importanza percettiva dei coefficienti DCT (*visually-weighted quantization*): più fine per le basse frequenze per cui il passo di quantizzazione deve essere piccolo, più grossolana invece per le alte frequenze dove si usano passi di quantizzazione molto ampi.

La quantizzazione è effettuata in pratica dividendo ogni coefficiente del blocco per il relativo passo di quantizzazione, il risultato viene poi arrotondato all'intero più vicino per ottenere gli indici relativi:

$$indici(i,j) = \left\lfloor \frac{y(i,j)}{q(i,j)} + \frac{1}{2} \right\rfloor$$

$\lfloor \alpha \rfloor$ rappresenta il più grande intero più piccolo di α, $y(i,j)$ è il coefficiente da quantizzare in posizione (i,j) all'interno del blocco 8×8, $q(i,j)$ il corrispondente passo di quantizzazione. In questo modo si associa ad ogni coefficiente da quantizzare l'indice dell'intervallo cui appartiene. In fase di decodifica viene recuperato il valore quantizzato con una inevitabile perdita di informazione, come mostrato nell'esempio in figura 29 e 30.

La matrice di quantizzazione può essere modificata in fase di codifica attraverso un parametro detto *fattore di qualità*. In questo modo è possibile scegliere se aumentare il livello di compressione, creando quindi maggiore distorsione nell'immagine, oppure migliorare la qualità dell'immagine, riducendo però il fattore di compressione. Infatti, scalando i valori della matrice, si ottiene un grado di libertà sulla quantità di informazione eliminata nell'immagine. Il fattore di qualità, L, può essere compreso tra 1 (molta informazione eliminata, immagine di pessima qualità) e 100 (pochissima informazione persa, immagine simile all'originale). La matrice è modificata nel seguente modo:

$$q_s(i,j) = \frac{(S \cdot q(i,j) + 50)}{100}$$

dove

$$S = \begin{cases} 5000/L & \text{se} \quad 0 < L < 50 \\ 200 - 2L & \text{se} \quad 50 \leq L < 100 \\ 1 & \text{se} \quad L = 100 \end{cases}$$

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Figura 28: Matrice di Quantizzazione usata nello standard JPEG.

$$\begin{bmatrix} 235.6 & -1.0 & -12.1 & -5.2 & 2.1 & -1.7 & -2.7 & 1.3 \\ -22.6 & -17.5 & -6.2 & -3.2 & -2.9 & -0.1 & 0.4 & -1.2 \\ -10.9 & -9.3 & -1.6 & 1.5 & 0.2 & -0.9 & -0.6 & -0.1 \\ -7.1 & -1.9 & 0.2 & 1.5 & 0.9 & -0.1 & 0.0 & 0.3 \\ -0.6 & -0.8 & 1.5 & 1.6 & -0.1 & -0.7 & 0.6 & 1.3 \\ 1.8 & -0.2 & 1.6 & -0.3 & -0.8 & 1.5 & 1.0 & -1.0 \\ -1.3 & -0.4 & -0.3 & -1.5 & -0.5 & 1.7 & 1.1 & -0.8 \\ -2.6 & 1.6 & -3.8 & -1.8 & 1.9 & 1.2 & -0.6 & -0.4 \end{bmatrix}$$

Figura 29: Coefficienti DCT corrispondenti ad un blocco di dati.

$$\begin{bmatrix} 15 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} 240 & 0 & -10 & 0 & 0 & 0 & 0 & 0 \\ -24 & -12 & 0 & 0 & 0 & 0 & 0 & 0 \\ -14 & -13 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figura 30: Indici corrispondenti al blocco di dati di fig.29 e relativo blocco DCT ricostruito.

6.3 Codifica entropica

Tutte le operazioni fatte finora hanno l'obiettivo di rendere più efficiente il processo di compattazione dati (codifica lossless), dove difatto avviene la riduzione dell'informazione da trasmettere e si produce la stringa di bit che rappresenta la sequenza di informazione. Come è possibile osservare dall'esempio mostrato in figura 30 tipicamente un blocco ha le seguenti caratteristiche: coefficiente DC non nullo, componenti a bassa frequenza diverse da zero, lunghe sequenze di zeri alle alte frequenze. La tecnica di compattazione scelta si adatta proprio alle proprietà del blocco per ottenere una codifica efficiente in termini di numero di bit spesi per la sua rappresentazione.

Innanzitutto lo standard codifica differentemente le etichette dei coefficienti DC e di quelli AC. Infatti, dalla (45) è evidente che la componente DC è un multiplo del valor medio di luminosità del blocco; poiché la luminosità media di un blocco non differisce sostanzialmente da quella dei blocchi adiacenti (quindi anche le etichette assumeranno valori molto vicini), ha senso usare una codifica predittiva lineare. Questo significa che viene codificata la differenza (errore di predizione) tra le etichette dei coefficiente DC di due blocchi consecutivi. In ogni caso il range di valori che può assumere l'errore di predizione è abbastanza ampio. Una codifica di Huffman[4] per un alfabeto così esteso sarebbe intrattabile dal punto di vista computazionale, per questo motivo si utilizza un particolare tipo di codifica caratterizzata dalle seguenti operazioni:

1. il range dei possibili valori che può assumere l'errore di predizione viene partizionato in intervalli, detti categorie, le cui ampiezze crescono secondo potenze di due (tabella 1):

[4]La codifica di Huffman si basa su un principio molto semplice ed intuitivo che è quello di associare lunghe stringhe di bit ai simboli meno frequenti, e sequenze più corte a quelli più frequenti.

Categoria	Ampiezza
0	0
1	-1, 1
2	-3, -2, 2, 3
3	-7, -6, -5, -4, 4, 5, 6, 7
4	-15,. . . , -8, 8,. . . , 15
5	-31,. . . , -16, 16,. . . , 31
6	-63,. . . , -32, 32,. . . , 63
7	-127,. . . , -64, 64,. . . , 127
8	-255,. . . , -128, 128,. . . , 255
9	-511,. . . , -256, 256,. . . , 511
10	-1023,. . . , -512, 512,. . . , 1023
11	-2047,. . . , -1024, 1024,. . . , 2047
12	-4095,. . . , -2048, 2048,. . . , 4095
13	-8191,. . . , -4096, 4096,. . . , 8191
14	-16383,. . . , -8192, 8192,. . . , 16383
15	-32767,. . . , -16384, 16384,. . . , 32767
16	32768

Tabella 1: Tabella Categoria e Ampiezza.

la categoria 0 ha come unico elemento lo zero, la categoria 1 ha due elementi (-1, 1), la categoria 2 ne ha quattro (-3, -2, 2, 3), e cosi via. Si noti come la classe di appartenenza indichi proprio il numero di bit necessari a rappresentare il valore del coefficiente in binario, infatti alla classe k appartengono esattamente 2^k possibili valori;

2. le categorie sono quindi codificate con l'algoritmo di Huffman facendo riferimento, per ognuna, alla distribuzione di probabilità media, trovata sperimentalmente dall'analisi di un ampio set di immagini; quindi per la codifica della categoria è sufficiente conoscere la tabella messa a disposizione dallo standard;

3. gli elementi di ogni categoria sono invece codificati con una stringa di bit a lunghezza fissa (se N sono gli elementi della categoria saranno necessari $\log_2 N$ bit).

Per rendere più efficace l'ultima fase della codifica lo standard JPEG utilizza una scansione a zig-zag della matrice (figura 31), che ha come obiettivo quello di riordinare i coefficienti posizionando quelli relativi alle basse frequenze (in prevalenza diversi da zero), prima di quelli delle alte frequenze (quasi tutti nulli). Aumenta in questo modo la possibilità di ottenere lunghe sequenze di zeri, che possono essere efficacemente codificate. In definitiva, per ogni blocco:

1. Il coefficiente DC verrà rappresentato dalla coppia (Categoria, Ampiezza), dove Categoria indica la classe di appartenenza, mentre Ampiezza è l'effettivo valore del coefficiente.

2. i coefficienti AC non nulli sono rappresentati dalla terna [(Run-Length, Categoria), Ampiezza], dove Categoria e Ampiezza hanno lo stesso significato di prima, mentre Run-Length è il numero di zeri consecutivi (da 0 a un massimo di 15) che si incontrano prima

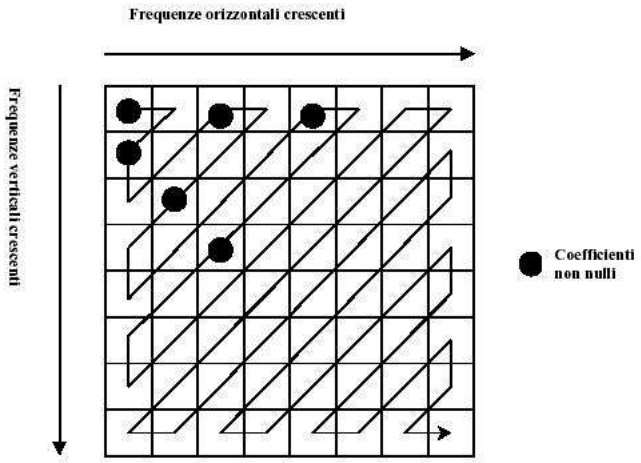
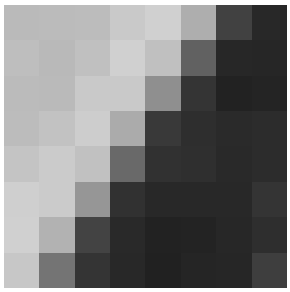


Figura 31: Scansione a zig-zag.



$$\begin{bmatrix} 187 & 188 & 189 & 202 & 209 & 175 & 66 & 41 \\ 191 & 186 & 193 & 209 & 193 & 98 & 40 & 39 \\ 188 & 187 & 202 & 202 & 144 & 53 & 35 & 37 \\ 189 & 195 & 206 & 172 & 58 & 47 & 43 & 45 \\ 197 & 204 & 194 & 106 & 50 & 48 & 42 & 45 \\ 208 & 204 & 151 & 50 & 41 & 41 & 41 & 53 \\ 209 & 179 & 68 & 42 & 35 & 36 & 40 & 47 \\ 200 & 117 & 53 & 41 & 34 & 38 & 39 & 63 \end{bmatrix}$$

Figura 32: Dati del blocco da comprimere.

del coefficiente corrente nella scansione. La coppia (Run-length, Categoria) è codificata in binario, mediante la tabella dello standard, l'Ampiezza con un codice a lunghezza fissa.

Inoltre nella fase di codifica bisogna tener conto dell'esistenza di due simboli speciali. Il primo simbolo è rappresentato dalla coppia (0,0), denominata EOB (End of Block), esso rappresenta la terminazione del blocco, si inserisce infatti quando nella scansione a zig-zag i rimanenti coefficienti da codificare sono tutti nulli. Il secondo simbolo è invece la coppia (15,0) che rappresenta 16 zeri consecutivi, la sua definizione è necessaria perché per non appesantire la complessità sono definite, nella tabelle di verità, stringhe di bit solo per i coefficienti AC preceduti al più da 15 zeri.

6.4 Esempio

Ripercorriamo adesso tutti i passi dell'algoritmo prendendo in considerazione un generico blocco dell'immagine. Supponiamo di avere a disposizione il blocco di dati 8×8 (estratto dall'immagine Lena) mostrato in figura 32: ogni pixel è codificato con 8 bit, quindi sono necessari 512 bit per memorizzarlo.

Dopo aver applicato la DCT-2D sui dati a media nulla si ottengono i coefficienti mostrati in figura 33.

$$\begin{bmatrix} -108.4 & 451.3 & 25.6 & -12.6 & 16.1 & -12.3 & 7.9 & -7.3 \\ 216.8 & 19.8 & -228.2 & -25.7 & 23 & -0.1 & 6.4 & 2 \\ -2 & -77.4 & -23.8 & 102.9 & 45.2 & -23.7 & -4.4 & -5.1 \\ 30.1 & 2.4 & 19.5 & 28.6 & -51.1 & -32.5 & 12.3 & 4.5 \\ 5.1 & -22.1 & -2.2 & -1.9 & -17.4 & 20.8 & 23.2 & -14.5 \\ -0.4 & -0.8 & 7.5 & 6.2 & -9.6 & 5.7 & -9.5 & -19.9 \\ 5.3 & -5.3 & -2.4 & -2.4 & -3.5 & -2.1 & 10 & 11 \\ 0.9 & 0.7 & -7.7 & 9.3 & 2.7 & -5.4 & -6.7 & 2.5 \end{bmatrix}$$

Figura 33: Coefficienti trasformati mediante DCT.

Realizziamo adesso la quantizzazione utilizzando la matrice di figura 33 e otteniamo le etichette mostrate in figura 34.

$$\begin{bmatrix} -7 & 41 & 2 & 0 & 0 & 0 & 0 & 0 \\ 18 & 1 & -16 & -1 & 0 & 0 & 0 & 0 \\ 0 & -5 & -1 & 4 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figura 34: Indici ottenuti con il processo di quantizzazione.

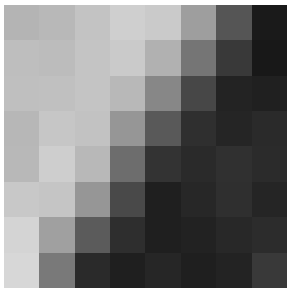
A questo punto si procede con la codifica dei valori secondo la scansione zig-zag. Si ottiene così la seguente sequenza di coppie e terne da codificare:

$(2,-2)\,(0,6,41)\,(0,5,18)\,(1,1,1)\,(0,2,2)\,(1,5,-16)\,(0,3,-5)\,(0,2,2)\,(2,1,-1)\,(0,1,-1)\,(3,3,4)$
$(1,1,-1)\,(5,1,1)\,(5,1,-1)\,(0,0)$

dove abbiamo tenuto conto che la DC del blocco precedente è -5, per cui l'errore di predizione è $-7+5=-2$, che appartiene alla categoria 2. Per codificare tale categoria la tabella di Huffman dello standard associa la parola 011, che sarà seguita dalla stringa 01 (codice a lunghezza fissa) necessaria a specificare che il valore da codificare è il secondo. Per i coefficienti AC bisogna usare la tabella che associa la codifica di Huffman per la coppia (Run-Length, Categoria) e poi una codifica a lunghezza fissa per l'Ampiezza. Per esempio, nella terna $(0,3,-5)$ alla coppia $(0,3)$ è associata la stringa 100, poi per specificare il valore -5 si trasmette 010 dato che è il terzo elemento della categoria. Procedendo per tutte le terne, si ottiene il seguente flusso di bit:

$011,01-111000,101010-11010,10010-1100,1-01,10-11111110110,01111-100,010-01,10-11100,0-00,0-111111110101,100-1100,1-1111010,1-1111010,0-1010$

Abbiamo ottenuto in questo modo una sequenza di 111 bit, quindi il tasso di codifica è diminuito a circa 1.7 bit/pixel. In figura 35 si mostra infine il blocco ricostruito.



$$\begin{bmatrix} 181 & 185 & 196 & 208 & 203 & 159 & 86 & 27 \\ 191 & 189 & 197 & 203 & 178 & 118 & 58 & 25 \\ 192 & 193 & 197 & 185 & 136 & 72 & 36 & 33 \\ 184 & 199 & 195 & 151 & 90 & 48 & 38 & 43 \\ 185 & 207 & 185 & 110 & 52 & 43 & 49 & 44 \\ 201 & 198 & 151 & 74 & 32 & 40 & 48 & 38 \\ 213 & 161 & 92 & 47 & 32 & 35 & 41 & 45 \\ 216 & 122 & 43 & 32 & 39 & 32 & 36 & 58 \end{bmatrix}$$

Figura 35: Blocco ricostruito.

7 Decodifica

Per ricostruire l'immagine dal file compresso è necessario realizzare le operazioni inverse a quelle eseguite nel processo di codifica, quindi è necessario:

1. riconoscere le stringhe che rappresentano la Categoria, l'Ampiezza e i simboli speciali, per ricostruire i blocchi 8×8;
2. moltiplicare i coefficienti per la tabella di quantizzazione utilizzata in fase di codifica;
3. realizzare la DCT inversa di ogni blocco e aggiungere la media ad ogni valore.

Notate che l'unica fase che causa distorsione è la 2. Ricordate poi che in fase di decodifica bisogna tener conto di eventuali colonne o righe di pixel di riempimento ed eliminarle dall'immagine.

8 Estensione alle immagini a colori

Le immagini non compresse sono generalmente rappresentate nello spazio RGB, tuttavia a causa delle particolari caratteristiche dell'occhio umano, molto più sensibile alle variazioni di luminosità che alle variazioni cromatiche, risulta conveniente rappresentare l'immagine nello spazio YC_bC_r, dove Y è la componente di luminanza e corrisponde proprio alla rappresentazione in scala di grigi dell'immagine stessa, mentre le componenti di crominanza C_b e C_r individuano il colore. In questo modo è possibile destinare maggiori risorse alla componente di luminosità. In effetti, la ricerca di uno spazio cromatico per l'immagine coincide con la ricerca di una rappresentazione in cui l'energia risulta compattata nella prima componente, alla quale vengono assegnati più bit in fase di compressione. Per passare dallo spazio RGB a quello YC_bC_r si usa la seguente trasformazione:

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.1687 & -0.3313 & 0.5 \\ 0.5 & -0.4187 & -0.0813 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

L'operazione di riduzione dell'informazione associata alle componenti di crominanza dell'immagine può essere realizzata prima ancora di fare ricorso all'algoritmo di compressione vero e proprio, sottocampionando le matrici delle componenti cromatiche di un fattore 2. Questa perdita di dati influisce poco sull'immagine a livello percettivo, proprio perché l'occhio umano è poco sensibile alle piccole variazioni cromatiche. E' anche chiaro allora che, a valle dell'algoritmo di codifica, le immagini a colori sono compresse di un fattore più elevato rispetto alle immagini monocromatiche. Nelle figure 36, 37 e 38 si mostrano alcuni esempi di immagini a colori codificate con lo standard JPEG con diversi livelli di qualità.

Riferimenti bibliografici

[1] A.Bovik, *The essential guide to Image Processing*, Academic Press, 2009.

[2] T.M.Cover and J.A. Thomas: *Elements of Information Theory*, Second Edition, Wiley Series in Telecommunications, 2005.

[3] A.Gersho and R.M.Gray: *Vector quantization and signal compression*. Boston: Kluwer Academic Publisher, 1992.

[4] K.Sayood: *Introduction to Data Compression*, Second Edition, Morgan Kaufmann 2000.

[5] C.E.Shannon: "A mathematical theory of Communication", *The Bell System Technical Journal*, vol.27, pp.379-423, 623-656, July, October, 1948.

Figura 36: Immagine originale 512×512 (a sinistra) e compressa con fattore di qualità $L = 50$ (a destra).

Figura 37: Immagine compressa con fattore di qualità $L = 10$ (a sinistra) e $L = 1$ (a destra).

Figura 38: Immagine originale (6.147 kB), compressa con fattore di qualità $L = 10$ (172 kB) e $L = 1$ (134 kB).