Alma Mater Studiorum · Università di Bologna

Scuola di Scienze
Corso di Laurea in Fisica

Trasformate Wavelet

Relatore:
Prof. Fabio Ortolani

Presentata da:
Lorenzo Gorini

Sessione II
Anno Accademico 2013/2014

Abstract

La presente tesi vuole dare una descrizione delle Trasformate Wavelet indirizzata alla codifica dell'immagine in formato JPEG2000. Dopo aver quindi descritto le prime fasi della codifica di un'immagine, procederemo allo studio dei difetti derivanti dall'analisi tramite la Trasformata Discreta del Coseno (utilizzata nel formato predecessore JPEG). Dopo aver quindi descritto l'analisi multirisoluzione e le caratteristiche che la differenziano da quest'ultima, analizzeremo la Trasformata Wavelet dandone solo pochi accenni teorici e cercando di dedurla, in una maniera più indirizzata all'applicazione. Concluderemo la tesi descrivendo la codifica dei coefficienti calcolati, e portando esempi delle innumerevoli applicazioni dell'analisi multirisoluzione nei diversi campi scientifici e di trasmissione dei segnali.

Indice

Introduzione

Negli ultimi anni i segnali digitali sono stati sfruttati in maniera sempre più massiccia. Essi sono presenti nelle immagini, nei video e nella musica. Per questo motivo, sono molto presenti in varie discipline come medicina, biologia, chimica e fisica. In quest'ultimo campo, ad esempio, le immagini vengono analizzate continuamente nell'ambito nucleare e subnucleare per identificare nuove particelle fondamentali tramite lo studio degli urti e delle traiettorie seguite. Un altro utilizzo della teoria dei segnali è quello che avviene nella Fisica della Materia quando sono osservati segnali elettrici provenienti da particolari materiali in analisi. In medicina, infine, la necessità di un efficiente sistema di immagazzinamento di immagini (radiografiche, tomografiche, ...) diventa sempre più importante.

In tempi relativamente recenti, grazie alla digitalizzazione dei segnali con la nascita dei computer, le analisi di immagini e video che una volta risultavano molto complesse e lunghe, ora sono divenute semplici e veloci grazie all'utilizzo di calcolatori sempre più potenti. E questo è importante in tutti i campi sopra elencati.

L'analisi dei segnali è, quindi, un campo di studio molto ampio e per non disperderci eccessivamente ci focalizzeremo sull'analisi dei segnali 2-D, cioè sulle immagini. Per evitare comunque una trattazione "astratta" inizieremo la nostra analisi tramite un esempio, del quale andremo successivamente ad analizzare le diverse fasi e le problematiche. Descriveremo, infatti, la creazione di un file JPEG: dallo scatto della macchina fotografica, fino ad arrivare alla compressione ed immagazzinamento di una sequenza binaria ottimizzata. Guarderemo poi nel dettaglio i concetti matematici che stanno alla base della compressione utilizzati nel formato JPEG, in particolare approfondiremo le trasformate di Fourier. Passeremo poi ad immaginare le caratteristiche di un algoritmo che possa superare i limiti intrinseci di queste trasformate e ottimizzi la compressione dell'immagine: arriveremo cioè a parlare delle wavelet e dei possibili vantaggi dell'analisi multirisoluzione.

In particolare l'approccio tipico dell'analisi multirisoluzione, permesso dalla Trasformata Wavelet, si basa sull'utilizzo di basi con specifiche (ma non troppo stringenti) proprietà che hanno la possibilità di essere ridefinite di volta in volta a seconda delle necessità (il tipo di analisi, la velocità o la qualità della compressione,...). In particolare le basi risultano essere delle funzioni della scala utilizzata e hanno la proprietà di essere

estremamente localizzate. Queste due caratteristiche permettono una suddivisione molto flessibile del piano tempo-frequenza, che risulta in questo modo molto utile a distinguere sia la frequenza della componente del segnale, ma anche il momento in cui è localizzata.

Per sviluppare e capire meglio le trasformate wavelet e le relative caratteristiche e potenzialità, prenderemo come punto di partenza la Trasformata Wavelet Continua, cioè la versione più ridondante della scomposizione tramite Wavelet. Procedendo poi verso l'eliminazione della ridondanza, quantizzeremo il piano tempo-frequenza in modo tale da ottenere una suddivisione nella quale i diversi rettangoli non si sovrappongono se non in minima parte. A questo punto per risolvere il problema di un numero di rettangoli divergente, lo analizzeremo da un punto di vista di sottoinsiemi dello spazio $l^2(R)$ a cui appartiene il segnale f(x). Studieremo cioè una suddivisione di quest'ultimo in spazi ortogonali generati dalle basi wavelet e da un altro tipo di funzioni, quelle di scala (o "funzioni scaling"). Tramite questi due tipi di funzioni di base scalabili e traslabili riusciremo a generare l'intero spazio $l^2(R)$ e, essendo quelli ortogonali, avremo identificato una base rispetto alla quale scomporre il segnale tramite la Trasformata Wavelet Discreta.

La trasformata Wavelet Veloce favorirà la comprensione della natura dell'analisi multirisoluzione tramite la codifica dell'immagine utilizzando le due sole funzioni-base, cioè quella wavelet e quella di scala. Ci renderemo conto in questo modo che, sfruttando solo queste due, potremo generare una serie di immagini a scale (cioè risoluzioni) differenti, ognuna delle quali contiene un numero di informazioni differenti. In questo modo avremo un ordinamento basato sulla qualità delle informazioni (cioè dei dettagli) che possono essere aggiunte all'immagine sottocampionata. Avremo quindi una suddivisione dei dettagli dell'immagine in base al loro grado di importanza, dalle strutture più macroscopiche fino ad arrivare a quelle più fini. Ed è questo il principio fondante dell'analisi multirisoluzione, ed è lo stesso principio su cui si basa la successiva codifica in formato JPEG2000 dei coefficienti ottenuti dalla Trasformata Wavelet Discreta.

L'algoritmo di codifica dei coefficienti provvederà quindi ad ordinarli in base al loro valore e relativa importanza per poter generare un flusso di dati che avrà la particolarità di poter essere tagliato in qualunque punto. E questa interruzione nella ricezione costituirà essa stessa un'ulteriore compressione del segnale poiché solo i primi coefficienti, quelli più significativi, saranno stati ricevuti e la decodifica genererà quindi un'immagine più compressa, con una qualità minore, ma comunque completa in ogni sua parte.

Capitolo 1

Analisi dei segnali

1.1 Dal mirino al file binario

Esistono vari tipi di immagini: tanti quanti sono i tipi di dispositivi in grado di produrle. Si parte infatti dalla comune macchina fotografica per arrivare a macchinari più evoluti e complessi come possono essere quelli all'interno di un ospedale (macchine per la risonanza magnetica, per la radiografia, per la TAC, ...), fino ai rivelatori di particelle dei grandi acceleratori in cui sono presenti gli elementi base che troviamo anche nella macchina fotografica come il sensore che riesce a quantificare il numero di particelle che lo attraversano.

Gli ultimi due tipi, ad esempio, producono immagini tridimensionali ma i meccanismi alla base dell'analisi di queste immagini sono comunque molto simili alle immagini bidimensionali di cui ci occuperemo.

Dopo aver inquadrato un soggetto, il fotografo preme il pulsante di scatto e la macchina fotografica alza lo specchio mostrato in figura 1.1 e contemporaneamente apre un foro più o meno grande, chiamato *diaframma*, che lascia passare la luce sul sensore. Quest'ultimo è composto da una matrice di *photosite* (cioè transistor CMOS che producono una coppia elettrone lacuna al passaggio di un fotone) ognuno sensibile ad una determinata lunghezza d'onda, tipica dei colori primari rosso, verde e blu.

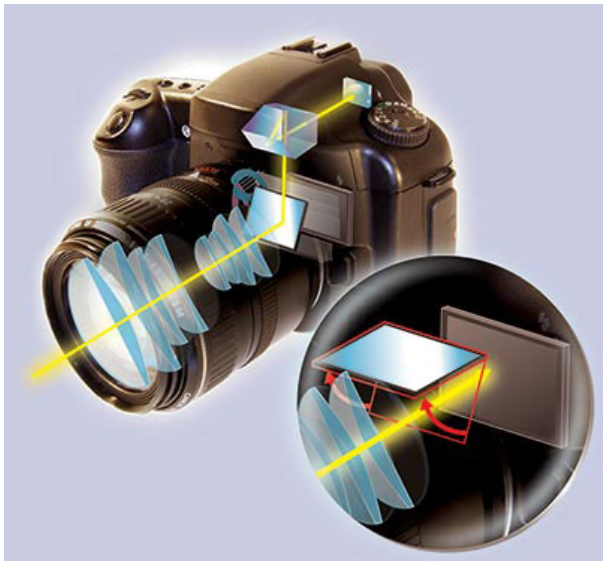


Figura 1.1: Rappresentazione schematica del percorso della luce in una macchina fotografica reflex. Possiamo notare che, mentre nella foto grande vediamo l'immagine riflettersi prima nello specchio e poi nel pentaprisma per arrivare al mirino, nel focus in basso vediamo lo specchio alzarsi e la luce giungere al sensore.

A questo punto il processore interno alla fotocamera memorizza una sequenza di bit contenente informazioni sul sensore: in relazione al numero di fotoni giunti sul singolo transistor passa un segnale quantizzato più o meno intenso. Come si può notare dalla figura 1.2, i sensori più comuni (di tipo *Color Filter Array*) contengono solitamente un numero diverso di transistor sensibili alle lunghezze d'onda dei tre colori primari; infatti al loro interno troviamo molti più *photosite* collegati al colore verde rispetto agli altri due: più precisamente il rapporto R:G:B è 1:2:1. Questo è dovuto al fatto che l'occhio umano è molto più sensibile a quella determinata lunghezza d'onda. I segnali analogici in uscita dai photodetector del sensore, dopo la loro conversione in digitale, possono andare in due direzioni diverse a seconda della impostazione della fotocamera: o verso il processore d'immagine interno che, attraverso l'algoritmo di interpolazione (demosaicizzazione), ricostruisce le due componenti mancanti su ogni photosite, oppure questi dati "grezzi "possono venire registrati appunto nel file Raw tramite un algoritmo *lossless* che spiegheremo in seguito. In questo secondo caso il singolo pixel conterrà l'informazione proveniente da ogni transistor separatamente: i colori primari non sono ancora stati elaborati per dare origine a tutti i colori della scala cromatica. Questo passaggio avviene in seguito quando il file viene trasferito ad un calcolatore (anche se in elaborazioni meno sofisticate può avvenire anche all'interno della stessa fotocamera sfruttando il processore e le impostazioni integrate). Per il momento quindi l'immagine ha bisogno di un'elaborazione ulteriore per essere visualizzata come quella in figura 1.3.

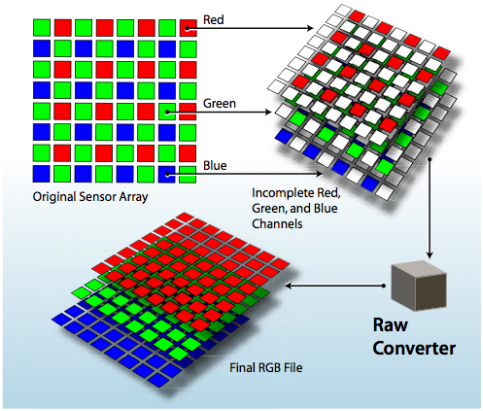


Figura 1.2: In quest'immagine possiamo notare le diverse fasi di ricostruzione dell'immagine a partire dalle informazioni del sensore. In particolare si può notare il rapporto RGB iniziale pari a 1:2:1, che viene modificato per generare un'immagine visualizzabile su schermi in cui il rapporto è 1:1:1.

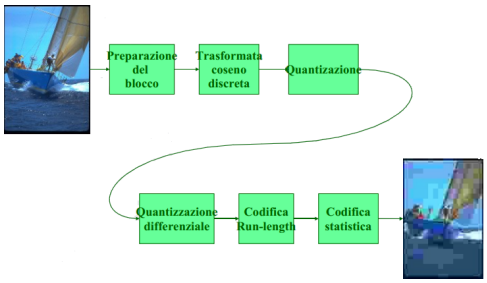


Figura 1.3: Procedura di codifica e decodifica di un'immagine JPEG.

Dopo il trasferimento al calcolatore, tramite opportuni programmi, si ha la possibilità di impostare in modo più preciso i parametri della fotografia come luminosità, contrasto, e anche modificare piccoli particolari. Quando l'immagine è definita, si passa alla delicata fase della compressione. In questa fase i pixel, che precedentemente contenevano un solo colore, vengono interpolati con i pixel vicini in modo da poter dare finalmente origine a tutta la scala cromatica. In realtà, avendo il calcolatore una memoria finita, questo non è possibile perché ogni pixel dovrà occupare un numero finito di *bit* e questo significa che, se scegliamo come *profondità di colore* 8 bit, l'intensità del singolo colore primario potrà essere compresa solamente tra 0 e 2^8-1 = 255 (figura 1.4). Scegliendo, però, come profondità di colore 16 bit, il file risulterà il doppio più grande dell'altro ma la limitatezza del *range* dei colori inciderà solo marginalmente sulla qualità della foto.

Una volta che i colori sono stati interpolati tramite algoritmi più o meno complessi (che non approfondiremo) e averli quantizzati, si passa alla trasformazione delle loro

componenti da RGB, che è una codifica estremamente ridondante, ad una nuova scala che permetterà errori minori nella compressione: quella YCbCr.

Figura 1.4: In quest'immagine notiamo le differenze derivanti da una scarsa *profondità di colore*. Si può notare come i colori si uniformino in chiazze omogenee e scompaiano le sfumature.

Questa sigla rappresenta la scomposizione dell'immagine nella componente della luminanza e nelle due della crominanza; le equazioni per passare dal sistema R'G'B' a quello YPbPr (versione analogica dello Y'CbCr) risultano le seguenti:

$$\begin{aligned} Y' &= K_r * R' + (1 - K_r - K_b) * G' + K_b * B', \\ P_b &= 0.5 * (B' - Y')/(1 - K_b), \\ P_r &= 0.5 * (R' - Y')/(1 - K_r), \end{aligned} \tag{1.1}$$

dove R', G', B' $\in [0; 1]$ sono le componenti già riscalate, Y' $\in [0; 1]$ si distingue dalla luminanza Y in quanto contiene già la *correzione di gamma*, Pb e Pr $\in [-0.5; 0.5]$. Kb e Kr sono dovute al fatto che l'YCbCr non è uno spazio colore assoluto, ma è un modo di codificare l'informazione RGB, e i colori realmente visualizzati (e quindi i parametri per correggerli) dipendono dai coloranti usati dal mezzo di visualizzazione. Quando questi risultati sono rappresentati nella forma digitale YCbCr, sono scalati e arrotondati, e di solito viene aggiunto un valore di offset.

YCbCr si riferisce alla codifica digitale di video e immagini, come per esempio negli schemi di compressione MPEG e JPEG, ed è concepito per rispecchiare il comportamento della visione umana scartando alcune informazioni di colore cui l'occhio umano non è molto sensibile. Buona parte del dettaglio percepito, infatti, è data dalle informazioni sui livelli luminosi presenti nel segnale di luminanza. Di conseguenza, i segnali Cb e Cr

possono essere compressi in maniera sostanziale ricorrendo al sottocampionamento della crominanza, dimezzando cioè la risoluzione delle informazioni di colore secondo lo schema 4:2:2. Essendo segnali di differenza cromatica, inoltre, è favorita una mascheratura più efficiente di eventuali errori di trasmissione o artefatti di compressione, rispetto a una rappresentazione RGB diretta.

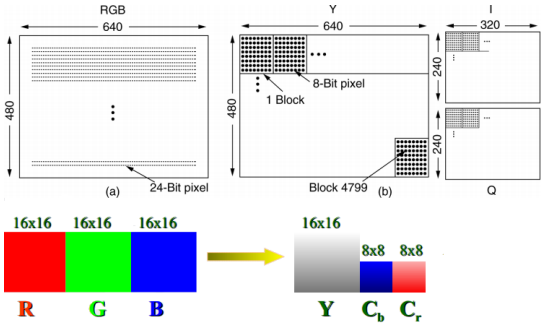


Figura 1.5: Nell'immagine (a) si può notare che i diversi pixel occupano ugualmente 8bit per ogni componente. Viceversa nell'immagine (b) si nota che solo la luminanza occupa 8 bit per ogni pixel, mentre le immagini contenenti i pixel dei Cb e Cr sono sottocampionate e quindi occupano solo un quarto della memoria rispetto alla luminanza

Dopo questa compressione minore, la fotografia ne subirà una molto maggiore grazie al formato JPEG. Questo formato, infatti, ha la capacità di ridurre notevolmente le dimensioni del file rinunciando solamente a dettagli meno significativi e lasciandone i contorni invariati. Nonostante il file sia composto da una sequenza di bit, esso rappresenta una matrice di pixel (ognuno codificato tramite le componenti YCbCr) e la compressione avverrà andando ad analizzare l'andamento dei contorni delle figure e codificando le diverse componenti come fossero tre diverse immagini.

Vedremo quindi nel prossimo paragrafo i concetti che sono alla base della compressione nel formato JPEG: le trasformate di Fourier.

1.2 Compressione e Ridondanza

Considereremo ora una sola delle componenti dei pixel del file che stiamo elaborando: abbiamo in questo modo un file contenente le diverse intensità. In seguito la stessa procedura potrà essere applicata alle restante componenti seguendo il medesimo procedimento.

Il termine *compressione di dati* definisce il processo che riduce la quantità di dati necessari a rappresentare una data quantità di informazione. Questo ovviamente è possibile grazie al fatto che parte dei dati sono ridondanti, nel senso che contengono informazioni

ripetute. Possiamo quantificare questa ridondanza R tramite la formula:

$$R = 1 - \frac{1}{C} \tag{1.2}$$

dove C è il rapporto di compressione definito come:

$$C = \frac{b}{b'} = \frac{numero\ di\ bit\ nell'immagine\ compressa}{numero\ di\ bit\ nell'immagine\ non\ compressa}. \tag{1.3}$$

Se quindi C=10, la ridondanza sarà 0,9 e questo significa che il 90% dei dati risulta ridondante. Questa ridondanza anche se eccessiva in questo caso può a volte avere risvolti positivi. Se, infatti, durante la trasmissione il segnale è soggetto a disturbi e a rumore, grazie al fatto che nel segnale non compresso gli stessi dati sono ripetuti dieci volte, possiamo riuscire a ricostruirlo individuando gli effetti del rumore ed eliminandolo. La ridondanza è quindi molto utile in ambienti particolarmente rumorosi come quelli industriali in cui la precisione e affidabilità del segnale è altrettanto importante.

Le matrici di intensità bidimensionali (come quelle dell'immagine rappresentante una singola componente) risentono di tre tipi principali di ridondanza dei dati:

- Ridondanza della codifica. Un codice è un sistema di simboli atti a rappresentare un insieme di eventi o una certa quantità di informazione. Il numero di bit che costituisce ciascun simbolo è la sua lunghezza. I codici a 8 bit, ad esempio, che vengono utilizzati per rappresentare la matrice contengono più bit del necessario, perché il numero di possibili valori che può assumere quel pezzo di informazione è minore di 2^8=256.

- Ridondanza spaziale e temporale. Dal momento che la maggior parte delle matrici che stiamo considerando contengono pixel relazionati spazialmente perché simili al loro intorno, l'informazione è replicata nei valori circostanti. Similmente in una sequenza video fotogrammi sono spesso molto simili tra loro.

- Informazione irrilevante. La maggior parte delle matrici contiene informazioni ignorate dal sistema visivo umano (dettagli troppo fini non rappresentabili dallo schermo o non visibili normalmente).

immagine i tre tipi della ridondanza dal gonzales

Gli algoritmi di compressione che da qui in avanti prenderemo in analisi avranno l'obiettivo di ridurre queste ridondanze cercando di evitare l'eliminazione di informazioni rilevanti. Perciò, mentre nell'ultimo caso la ridondanza, una volta identificata tramite studi dell'apparato visivo umano e tramite l'analisi dei dettagli eccedenti dell'immagine, è eliminabile in blocco, gli altri due tipi di ridondanza saranno quelli che richiederanno algoritmi più complessi. Sarà importante infatti distinguere le informazioni ridondanti da quelle necessarie per non rischiare di eliminare dettagli dell'immagine.

Affrontiamo dapprima la ridondanza della codifica.

1.3 Algoritmi *Lossless*

Gli algoritmi di compressione possono essere di due tipi: *lossless* (cioè senza perdita di qualità) oppure *lossy* (cioè con perdita). Questo significa che, mentre i primi hanno la capacità di ridurre leggermente la dimensione del file e permettono il recupero della sequenza dei bit di partenza in modo esatto, i secondi possono comprimere il file notevolmente ma il processo non è reversibile. A seconda degli utilizzi e necessità si utilizza uno, l'altro, o anche una combinazione dei due.

Ad esempio nel caso del formato JPEG che stiamo prendendo in esame, inizialmente si procede comprimendo il file con un algoritmo *lossy* chiamato *Trasformata Discreta del Coseno* (o DCT), e successivamente si utilizza la compressione *lossless* per ridurre il file ulteriormente.

Una delle tecniche *lossless* più famose che possiamo utilizzare per la rimozione della ridondanza nella codifica si deve a Huffman. Quando si codificano i simboli di una sorgente di informazione in modo singolo, la codifica di Huffman ottiene il più piccolo numero possibile in termini di lunghezza di codice (numero di bit) per ogni simbolo della sorgente.

L'idea che sta alla base dell'algoritmo è quella di codificare con un numero di bit minore, i simboli che compaiono più frequentemente: in questo modo ogni volta che compariranno potrò scriverli in forma abbreviata riducendo la lunghezza del codice. Le fasi per la codifica sono le seguenti. Il primo passo consiste nel creare una serie di riduzioni della sorgente ordinando le probabilità dei simboli, combinando quelli meno frequenti e assegnandogli un unico simbolo. In questo modo posso assegnare al mio insieme di simboli (che a questo punto hanno probabilità molto simile) un codice della lunghezza necessaria affinché siano tutti diversi tra loro. Avendo ora stabilito la codifica per i simboli più frequenti (anche se raggruppati), vado ad identificare i gruppi con probabilità più alta e vado ad allungare leggermente il codice per i simboli più frequenti (che eventualmente corrispondono ancora ad una molteplicità di simboli). Proseguendo in questo processo arrivo ad avere un albero come quello in figura 1.6 con diversi nodi da cui partono sempre e solo due rami ai quali verrà assegnato uno 0 o un 1 per distinguerli. Dopo un certo numero di nodi trovo uno e un solo simbolo, identificato dal codice che ottengo unendo in una sequenza il valore binario di ogni ramo che devo percorrere per arrivare ad esso.

A questo punto, andando a codificare la sequenza di bit dell'immagine, potrò sostituire i simboli con il codice specificato dal "dizionario" descritto. Se ad esempio avessi una sequenza in cui si alternano unicamente due simboli, mi basterebbe rappresentarli con 0 e 1, senza dover scrivere ogni volta l'intera rappresentazione binaria classica dei due simboli (e.g. ASCII).

Questo metodo prettamente matematico può quindi essere applicato ad una qualunque sequenza di bit, e a qualunque tipo di file. Ne sono un esempio i file .RAR, .ZIP,... Nel caso della foto introdotta nel paragrafo 1.1 questi algoritmi verranno utilizzati al

termine della compressione "Lossy", che descriveremo nel prossimo paragrafo, per ridurre ulteriormente e per l'ultima volta la dimensione del file.

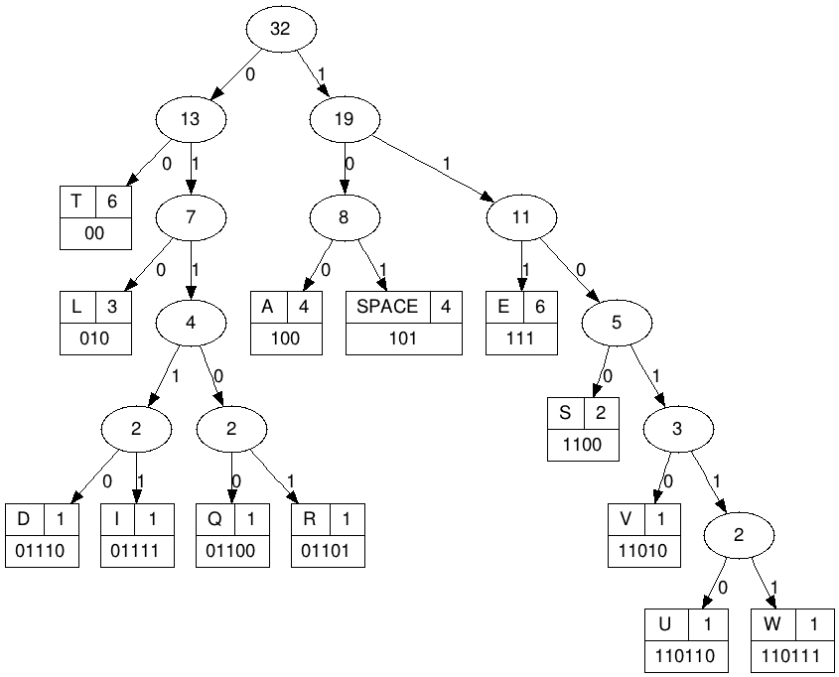


Figura 1.6: La codifica di Huffman della frase "Questa tesi tratta delle Wavelet" sopra generato dal programma on-line "Huffman Tree Generator" di Slawek Ligus (2010). L'algoritmo in questo caso genera una *lunghezza media del codice* pari a: $\sum_{n=1}^{N}(\textit{probabilità del simbolo } n \ \in [0,1]) * (\textit{numero di bit utilizzati per il simbolo})) = 3,375$; il risultato rappresenta in questo caso il numero di bit che in media dovrò utilizzare per codificare un codice in cui le probabilità delle singole *codeword* sono quelle, come succede infatti quando codifico la frase suddetta. è quindi espresso come $\frac{bit}{elemento\ della\ matrice}$,cioè $\frac{bit}{componente\ del\ singolo\ pixel}$.

1.4 Compressione *Lossy*

Affrontiamo ora la ridondanza spaziale, anche se quella temporale si basa sugli stessi concetti. Dal momento che la matrice bidimensionale che stiamo considerando contiene pixel correlati spazialmente perché spesso simili ai valori circostanti, cercheremo di codificare con un unico simbolo i pixel che si ripetono, come potrebbero essere quelli che rappresentano uno sfondo omogeneo.

Questa procedura comporta ovviamente una perdita di informazione (*lossy*) sulle minime sfumature dello sfondo ma cercheremo di mantenere un buon rapporto tra la

compressione e la qualità dell'immagine. Quest'ultima potrà essere scelta da parte dell'utente, una volta definito l'algoritmo ottimizzato, in base alla dimensione necessaria per i diversi utilizzi: dalla pagina web, in cui il file deve essere particolarmente leggero per favorirne il caricamento, fino alla stampa di un poster, che richiede un immagine ricca di dettagli. Oltre alle risorse computazionali che si hanno a disposizione, questo rapporto è infatti la caratteristica che si deve prendere in considerazione quando si sviluppano i diversi algoritmi.

Questa tecnica divide l'immagine in piccoli blocchi non sovrapposti di uguale dimensione (per esempio 8 x 8) ed elabora i blocchi in modo indipendente utilizzando una trasformata in 2-D. In questa particolare codifica, la trasformata viene utilizzata per mappare ogni *blocco* o *sottoimmagine* in un insieme di coefficienti della trasformata, che vengono successivamente quantizzati e codificati. In particolare si sfrutta la possibilità riscontrata nella maggior parte delle immagini di quantizzare in modo grossolano, o addirittura scartare, tutti i coefficienti con una piccola magnitudo, introducendo solo una piccola distorsione dell'immagine. Proprio per trasformare i dati dell'immagine, può essere utilizzata una grande varietà di trasformate, in modo tale da minimizzare i coefficienti più piccoli che verranno scartati e poter concentrare tutta l'informazione nei coefficienti maggiori che verranno memorizzati. In generale si consideri la *sottoimmagine* g(x,y) di dimensioni n x n la cui trasformata discreta, T(u,v), può essere espressa in termini della relazione generale:

$$T(u,v) = \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} g(x,y)\, r(x,y,u,v) \tag{1.4}$$

per u,v = 0,1,2,...,n-1. Viceversa, data T(u,v), g(x,y) può essere ottenuta tramite la trasformata inversa discreta:

$$g(x,y) = \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} T(u,v)\, s(x,y,u,v) \tag{1.5}$$

per x,y = 0,1,2,...,n-1.

In queste equazioni, r(x,y,u,v) ed s(x,y,u,v) vengono chiamati rispettivamente "kernel della trasformata" e "kernel della trasformata inversa". Dal momento che queste due funzioni dipendono solamente dagli indici x,y,u,v e non dai valori g(x,y) e T(u,v), definisce un insieme di "funzioni di base" o di "immagini di base" per le due serie che abbiamo definito. In particolare possiamo modificare la notazione utilizzata sfruttando la forma matriciale:

$$\mathbf{G} = \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} T(u,v)\, \mathbf{S}_u v \tag{1.6}$$

dove **G** è una matrice contenente i pixel di g(x,y) e

$$S_{uv} = \begin{bmatrix} s(0,0,u,v) & s(0,1,u,v) & \cdots & s(0,n-1,u,v) \\ s(1,0,u,v) & \ddots & \cdots & \vdots \\ \vdots & \cdots & \ddots & \vdots \\ \vdots & \cdots & \ddots & \vdots \\ s(n-1,0,u,v) & s(n-1,1,u,v) & \cdots & s(n-1,n-1,u,v) \end{bmatrix} \tag{1.7}$$

dove G viene esplicitamente definita come una combinazione lineare di n^2 matrici di dimensioni n x n, cioè le $S_{u,v}$ per u,v = 0, 1, 2, ..., n-1. Queste matrici sono dunque le "immagini" (o, più propriamente, "funzioni") "di base" dell'espansione in serie PRECEDENTE(QUELLA 8.2-20). La semplificazione che si ottiene ponendo a zero i coefficienti minori, quindi, si fonda sulla natura ortonormale delle immagini della base e questo significa che l'approssimazione totale dell'errore, e della perdita di informazione, è pari alla somma dei coefficienti che poniamo a zero. Nel paragrafo seguente illustreremo a livello intuitivo le basi teoriche che permettono l'analisi di una funzione appartenente ad l^2(Z) tramite uno sviluppo in serie. Prima di affrontarle, però, notiamo che la restrizione imposta dall'appartenenza di una funzione allo spazio l^2(Z) non è particolarmente stringente. La definizione di appartenenza ad l^2(Z) è basata infatti sulla finitezza della norma della sequenza x_n:

$$||x||_2 = \left(\sum_{n=-\infty}^{\infty} |x_n|^2 \right)^{1/2} < \infty \tag{1.8}$$

.

Questo fisicamente vuol dire che la sequenza dovrà essere generata da un segnale di energia finita, e questa è una condizione facile da soddisfare per qualunque segnale che si protrae per un tempo finito.

1.4.1 Basi teoriche della trasformata di Fourier

Quando utilizziamo la Trasformata di Fourier, allo stesso modo della DCT, stiamo sfruttando il fatto che l^2(Z) rappresenti ed abbia tutte le caratteristiche di uno spazio di Hilbert, la cui definizione risulta essere la seguente.

Definizione 1 *Uno spazio vettoriale H è detto spazio di Hilbert se verifica le seguenti condizioni:*

1. *H è uno spazio pre-hilbertiano (ovvero in H è definito un prodotto scalare <x,y> tra due arbitrari vettori x, y di H).*

2. *H è completo come spazio metrico con la metrica indotta dalla norma, a sua volta indotta dal prodotto scalare:* $d(x,y) = |x-y| = sqrt(< x-y, x-y >)$,

   cioè H ha la struttura di spazio di Banach (ovvero di uno spazio lineare normato completo).

Una volta definiti, quindi, il prodotto scalare e una metrica in uno spazio vettoriale completo, posso essere sicuro che esiste una base ortonormale dello spazio di Hilbert $l_2(Z)$ secondo il teorema molto generale che segue.

Teorema 1 *Ogni spazio vettoriale non nullo X ammette una base. Ogni sistema di vettori linearmente indipendenti, se non forma una base può essere completato in modo da formare una base*

A questo punto tralasceremo la dimostrazione del fatto che la base scelta

$$\varphi_n(x) = \frac{1}{\sqrt{2\,\pi}}\, e^{i\,n\,x} \tag{1.9}$$

Per qualunque n $\in$ Z, vada a formare un sistema ortonormale completo nello spazio $l^2(Z)$. Conoscendo quindi la base ortonormale possiamo utilizzare un altro teorema per capire meglio le serie di Fourier.

Teorema 2 *Sia* $\{x_\alpha\}_{\alpha\,\in\,A}$ *un sistema ortonormale in uno spazio di Hilbert H. Allora le due affermazioni sono equivalenti.*

1. $\{x_\alpha\}_{\alpha\,\in\,A}$ *è una base ortonormale.*

2. *Per ogni* $x \in H$ *vale la seguente relazione di Parseval:*

$$|x|^2 = \sum_\alpha |< x_\alpha, x >|^2, \tag{1.10}$$

dove la somma, come nel caso precedente, risulta estesa al più ad una infinità numerabile di termini non nulli.

Già da questo teorema si può notare la comparsa della somma delle proiezioni per andare a ricostituire la norma della funzione. Infatti in questo modo abbiamo mostrato, almeno a livello intuitivo, che, grazie all'ortonormalità e alla completezza del sistema delle basi $\varphi_n(x)$, possiamo dire che la somma delle proiezioni della funzione sui sottospazi generati dalle basi converge alla funzione secondo la norma di $l^2(Z)$.

Tornando quindi alle sottoimmagini che stavamo considerando, esprimiamo lo sviluppo in serie tramite la notazione matriciale.

1.4.2 Compressione nel formato JPEG

Nel caso del formato JPEG, ad esempio, si utilizza la Trasformata Discreta del Coseno (abbreviata come DCT). Questa è lineare e reversibile e ha le sue basi teoriche nella trasformata di Fourier, che infatti è capace di scrivere una qualunque funzione, che soddisfi particolari caratteristiche, come una serie di seni e coseni. Questa trasformata è ottenuta sostituendo i kernel che seguono nelle equazioni (1.4) e (1.5) :

$$\begin{aligned} r(x,y,u,v) &= s(x,y,u,v) \\ &= \alpha(u)\alpha(v)\ \cos\left[\frac{(2x+1)u\pi}{2n}\right]\ \cos\left[\frac{(2y+1)v\pi}{2n}\right] \end{aligned} \tag{1.11}$$

dove

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{n}}\ per\ u\ =\ 0 \\ \sqrt{\frac{2}{n}}\ per\ u\ =\ 1,2,\ldots,n-1 \end{cases} \tag{1.12}$$

e analogamente per α(v). La figura 1.7 mostra r(x,y,u,v) per n=8.

Nella pratica, nel formato JPEG la compressione avviene in tre "mosse": calcolo della DCT, quantizzazione, e infine assegnazione del codice a lunghezza variabile. Come abbiamo detto, l'immagine viene inizialmente partizionata a blocchi di dimensione 8 x 8.

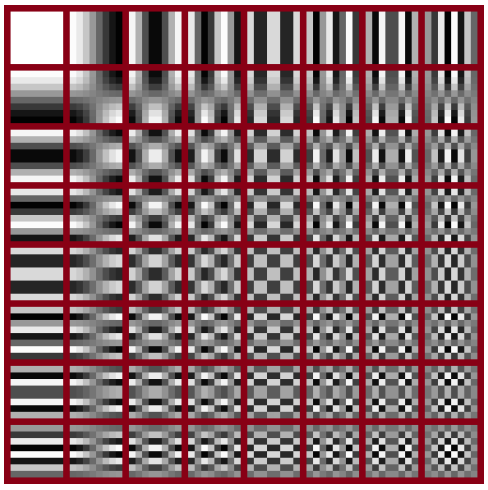


Figura 1.7: Tabella con le prime 64 funzioni di base della DCT limitata alle sottoimmagini 8x8. L'origine di ciascun blocco è in alto a sinistra. I valori di intensità più luminosi corrispondono a quelli di r più grandi. La frequenza u aumenta scendendo sull'asse verticale a partire da u=0, mentre v aumenta spostandosi verso destra.

Viene quindi calcolata la trasformata discreta del coseno 2-D del blocco, che viene a sua volta quantizzata e riordinata utilizzando un particolare modello per formare una

sequenza 1-D dei coefficienti quantizzati. Queste due ultime fasi avvengono a causa della finitezza della memoria del calcolatore: un coefficiente in virgola mobile avrebbe infatti una rappresentazione binaria più complessa di un numero intero. Per questo motivo le immagini trattano sempre con numeri interi e solitamente il range di possibili valori è quello che può essere contenuto in un numero finito di bit come 8,10,12 o, per esigenze professionali, anche 16 bit. Quindi il range di valori nel caso degli 8 bit è $[-2^7; 2^7 - 1]$. Perciò considerando una sottoimmagine con 64 pixel di intensità differenti, essendo già stati memorizzati come numeri interi, dobbiamo solo traslarli sottraendogli una quantità pari a $2^7 = 128$. Avendo ottenuto un vettore 1-D con 64 elementi che rappresentano i coefficienti rispetto ai vettori di base della DCT, la procedura di codifica trae vantaggio dalle lunghe sequenze di zeri che vengono normalmente create dal riordinamento.

Esempio 1 *[2] Si consideri la compressione e la ricostruzione della seguente sottoimmagine 8 x 8 con lo standard baseline JPEG:*

$$\begin{vmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 66 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 63 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \end{vmatrix}$$

Come prima operazione traslo tutti i valori sottraendo $2^7 = 128$*:*

$$\begin{vmatrix} -76 & -73 & -67 & -62 & -58 & -67 & -64 & -55 \\ -65 & -69 & -62 & -38 & -19 & -43 & -59 & -56 \\ -66 & -69 & -60 & -15 & 16 & -24 & -62 & -55 \\ -65 & -70 & -57 & -6 & 26 & -22 & -58 & -59 \\ -61 & -67 & -60 & -24 & -2 & -40 & -60 & -58 \\ -49 & -63 & -68 & -58 & -51 & -65 & -70 & -53 \\ -43 & -57 & -64 & -69 & -73 & -67 & -63 & -45 \\ -41 & -49 & -59 & -60 & -63 & -52 & -50 & -34 \end{vmatrix}$$

Dopo aver applicato la trasformata DCT delle equazioni (1.4) e (1.11) per n=8 diventa:

$$\begin{vmatrix} -415 & -29 & -62 & 25 & 55 & -20 & -1 & -3 \\ 7 & -21 & -62 & 9 & 11 & -7 & -6 & 6 \\ -46 & 8 & 77 & -25 & -30 & 10 & 7 & -5 \\ -50 & 13 & 35 & -15 & -9 & 6 & 0 & 3 \\ 11 & -8 & -13 & -2 & -1 & 1 & -4 & 1 \\ -10 & 1 & 3 & -3 & -1 & 0 & 2 & -1 \\ -4 & -1 & 2 & -1 & 2 & -3 & 1 & -2 \\ -1 & -1 & -1 & -2 & -1 & -1 & 0 & -1 \end{vmatrix}$$

Usando poi una specifica tabella di normalizzazione, riscalo tutti i valori in modo tale da ridurre il valore massimo e poter porre a zero buona parte dei coefficienti più piccoli. Ottengo quindi la matrice:

$$\begin{vmatrix} -26 & -3 & -6 & 2 & 2 & 0 & 0 & 0 \\ 1 & -2 & -4 & 0 & 0 & 0 & 0 & 0 \\ 3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ 4 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{vmatrix}$$

Dove per esempio il coefficiente

$$\hat{T}(0,0) = round\left(\frac{T(0,0)}{Z(0,0)}\right) = round\left(\frac{-415}{16}\right) \simeq -26 \tag{1.13}$$

usando la matrice di normalizzazione

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

Fino a questo momento abbiamo semplicemente descritto la compressione di un'immagine in formato JPEG. Ora ci addentreremo nelle problematiche e nelle caratteristiche

più teoriche di questo formato per andare a delineare i procedimenti che possono essere migliorati.

1.5 Verso un nuovo formato...

Analizziamo ora da un punto di vista matematico le procedure che abbiamo utilizzato per comprimere il file JPEG e ci focalizziamo in particolare sulle caratteristiche della Trasformata Discreta del Coseno (DCT). Come abbiamo già ripetuto, la DCT è una trasformata simile alla trasformata discreta di Fourier (DFT), ma fa uso solo di numeri reali. È equivalente a una DFT di lunghezza circa doppia, che opera su funzioni reali e pari (dato che la trasformata di Fourier di una funzione reale e pari è reale e pari a sua volta), dove in alcune varianti l'input e/o l'output sono traslati di mezzo campione. Una trasformazione correlata è la trasformata discreta del seno (DST), che è equivalente ad una DFT su numeri reali e funzioni dispari. [1]

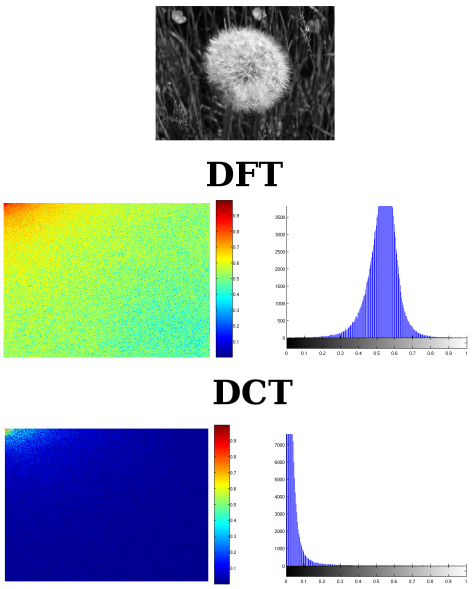


Figura 1.8: Confronto tra la Trasformata Discreta di Fourier (DFT) e la Trasformata Discreta del Coseno (DCT tipo II) di un'immagine. Da quest'immagine si può notare chiaramente il motivo per il quale la DCT (tipo II) sia preferibile nella compressione dell'immagine; lo spettro della DFT, infatti, risulta notevolmente più diffuso dello spettro della DCT e per questo motivo porre a zero i coefficienti minori comporterebbe una grossa perdita d'informazioni sull'immagine. La DCT concentra le informazioni nelle basse frequenze.

Un'altra differenza tra le due trasformate è la richiesta di periodicità, che è necessaria in quanto il segnale è riprodotto su un dominio infinito di cui, solo in un secondo momento, si considera solo una piccola porzione. Perciò, mentre la DFT richiede una periodicità di 2π in cui l'immagine si ripete, e questo, nel caso delle sottoimmagini 8x8, genererebbe grosse discontinuità lungo i bordi e quindi all'interno del segnale il cui dominio è in realtà infinito, la DCT, per ragioni teoriche correlate alle condizioni al contorno, richiede un'estensione pari della funzione (e quindi della sottoimmagine) e questo elimina le discontinuità nel segnale. Ad eccezione di queste differenze, la teoria che sta alla base di entrambe le trasformate è molto simile e per questo motivo studieremo la DCT al pari della Trasformata Discreta di Fourier. [8]

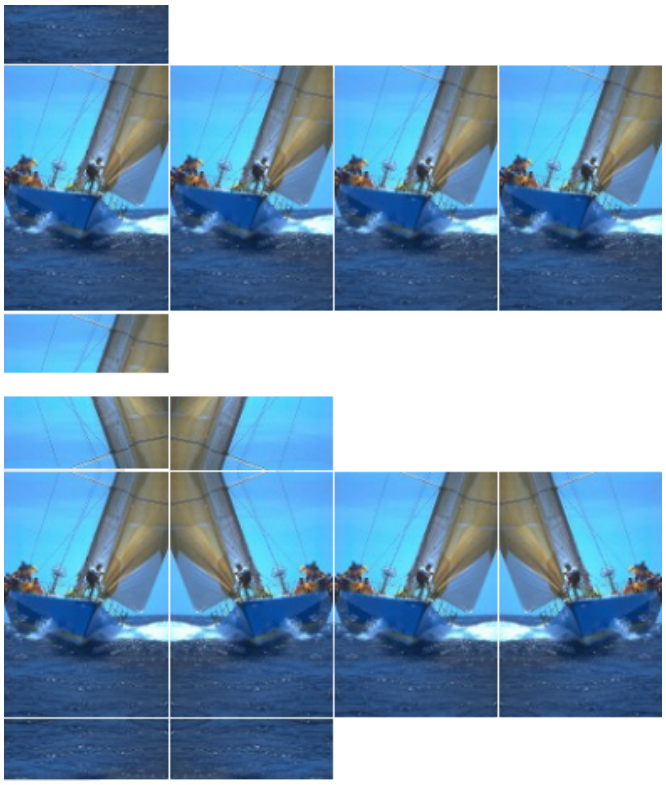


Figura 1.9: Confronto tra le condizioni al contorno richieste dalle due trasformate. Mentre nel caso della DFT si riscontrano forti discontinuità tra le immagini, nel caso della Trasformata del Coseno i bordi dell'immagine rimangono continui consentendo un'analisi più accurata e precisa.

Il primo svantaggio, forse il più importante, che riscontriamo parlando dell'espansione di Fourier è quello di avere risoluzione solo in frequenza e non nel tempo (o nello spazio, nel caso delle immagini). Il significato sarà più chiaro dopo aver definito le *deviazioni*

standard di una sequenza. Infatti quest'ultima, nella forma normalizzata $\frac{|x(e^j w)|^2}{||x||^2}$, può essere interpretata come *Funzione Densità di Probabilità* di una variabile casuale[6].

Definizione 2 *Data una sequenza $x_n \in l^2(Z)$ definiamo*

$$\triangle_t = \left(\frac{1}{||x||^2} \sum_{n \in \boldsymbol{Z}} (n - \mu_t)^2 \, |x_n|^2 \right)^{1/2} \tag{1.14}$$

dove

$$\mu_t = \frac{1}{||x||^2} \sum_{n \in \boldsymbol{Z}} n \, |x_n|^2 \quad . \tag{1.15}$$

Chiameremo $\triangle_t$ la deviazione standard della sequenza nel tempo.

Similmente possiamo definire la *deviazione standard* nel dominio delle frequenze utilizzando come sequenza la Trasformata della sequenza originale $\frac{|X(e^j w)|^2}{||X||^2}$.

Definizione 3 *Data una sequenza $x_n \in l^2(Z)$ definiamo*

$$\triangle_f = \left(\frac{1}{2\,\pi\,||x||^2} \int_{-\pi}^{\pi} (\omega - \mu_f)^2 \, |X(e^{j\,\omega}|^2 d\omega \right)^{1/2} \tag{1.16}$$

dove

$$\mu_f = \frac{1}{2\,\pi\,||x||^2} \int_{-\pi}^{\pi} \omega \, |X(e^{j\,\omega}|^2 d\omega. \tag{1.17}$$

Chiameremo $\triangle_f$ la deviazione standard della sequenza in frequenza.

Quando diciamo che la risoluzione nel tempo è nulla, il significato a questo punto è più chiaro: la deviazione standard delle sequenze basi è nulla; sappiamo, infatti, che le basi sono somme di seni e coseni che si ripetono periodicamente su tutto lo spazio. Un altro modo per spiegarlo è dire che, anche se possiamo scegliere con esattezza la frequenza della base, non possiamo conoscere dove quel particolare coefficiente e quella particolare base sia presente all'interno dell'immagine poiché è presente in tutta. Un esempio è quello della fig.(1.5) in cui si può notare che la base è localizzata in tutta la sottoimmagine, mentre la frequenza è quella definita dai coefficienti u e v. Per evitare il problema che abbiamo appena spiegato, sono state ideate diverse soluzioni negli ultimi decenni con l'obiettivo di avere una risoluzione soddisfacente sia nel dominio della frequenza che in quello dello spazio. L'idea che sta dietro a queste rappresentazioni nel dominio del tempo e della frequenza è quello di tagliare il segnale di interesse in diverse parti e analizzarle separatamente, in modo molto simile a quello che avviene con la Trasformata Discreta del Coseno applicata alle sottoimmagini 8 x 8. In quel caso infatti si parla a volte di Trasformate di Fourier Locali che in parte risolvono il problema o perlomeno cercano di

proporre una soluzione. È chiaro infatti che questo procedimento fornisce più informazioni sulla località delle differenti componenti, ma sorge un problema fondamentale: come tagliare il segnale? Supponiamo infatti di voler conoscere in modo assolutamente esatto tutte le componenti della frequenza presenti in un preciso istante. Tagliamo quindi una finestra di tempo infinitesima del segnale sfruttando la Delta di Kronecker, ne calcoliamo la trasformata, e scopriamo che qualcosa nel nostro ragionamento è andato storto. Il problema è che tagliare il segnale corrisponde ad una convoluzione tra il segnale e la finestra di taglio. Sfruttiamo a questo punto un'importante proprietà della Trasformata di Fourier.

Teorema 3 *Se f e g appartengono a* $l^1(R^n)$*, allora vale la relazione:*

$$F(f * g) = (2\,\pi)^{n/2}\,(F(f) \cdot F(g)). \tag{1.18}$$

Poiché, quindi, la convoluzione nel tempo è identica alla moltiplicazione nel dominio delle frequenze, sapendo che la trasformata della Delta di Kronecker è 1 in tutto il dominio, risulta evidente che il segnale sarà omogeneo e non localizzabile in frequenza.

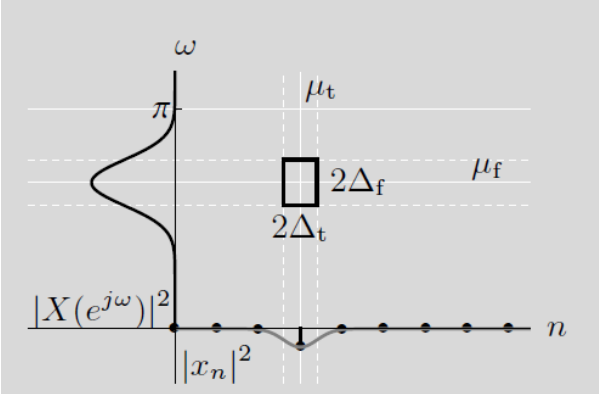


Figura 1.10: Piano tempo-frequenza. La sequenza x con DTFT X ha un *rettangolo di Heisenberg* centrato in (μ_t, μ_f) di larghezza $2\,\triangle_t$ ed altezza $2\,\triangle_f$

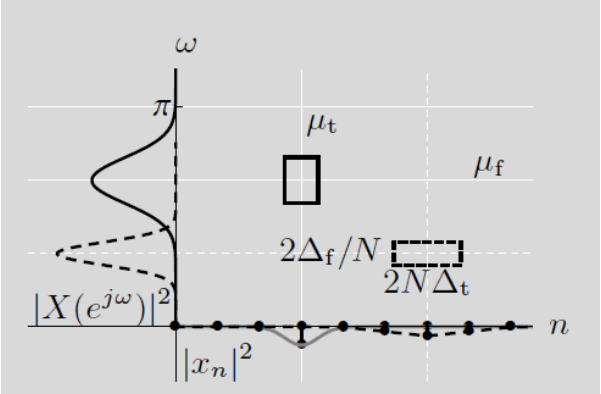


Figura 1.11: Un esempio delle possibili operazioni sulla sequenza. Si può notare, però, che l'area del *rettangolo di Heisenberg* rimane invariata. In particolare la trasformazione sposta il *rettangolo di Heisenberg* nelle coordinate (μ_t, μ_f) e ne riscala la larghezza al valore $2\,N\,\triangle_t$ e l'altezza al valore $\frac{2\triangle_f}{N}$. (Illustrato per N = 2)

Il principio che sta alla base del ragionamento appena descritto è dovuto al principio d'indeterminazione di Heisenberg applicato all'analisi dei segnali, il quale stabilisce che, come si può vedere dalla figura 1.12, è impossibile conoscere con esattezza la frequenza e conoscere contemporaneamente anche il tempo in cui è presente nel segnale. [1] In linguaggio più rigoroso, considerando il piano tempo-frequenza mostrato in figura 1.10, possiamo scrivere i seguenti due teoremi che non andremo a dimostrare:

Teorema 4 *Sia $x \in l^2(Z)$ avente deviazione standard nel tempo $\triangle_t$, e nella frequenza $\triangle_f$ e la sua Trasformata Discreta di Fourier soddisfi $X(e^{j\pi}) = 0$. Allora,*

$$\triangle_t \cdot \triangle_f > 1/2 \tag{1.19}$$

Chiameremo il rettangolo di lato $2\triangle_t$ e $2\triangle_f$ Heisenberg box (o rettangolo di Heisenberg)

Teorema 5 *Sia x una sequenza non nulla $\in \boldsymbol{C}^N$ con N_t valori non nulli, e sia X la sua Trasformata Discreta di Fourier X con N_f valori non nulli. Allora,*

$$N_t \cdot N_f \geq N \tag{1.20}$$

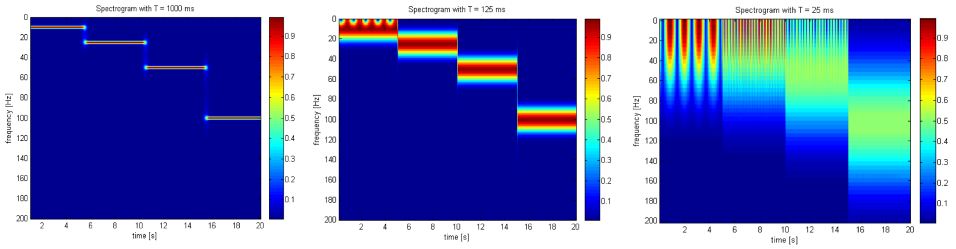


Figura 1.12: Spettrogrammi analizzati con finestre temporali di 25,125 e 1000 ms di un segnale emesso a frequenze diverse mai sovrapposte. Si può notare che, mentre nella prima immagine, in cui la finestra di tempo in cui viene analizzato il segnale è maggiore, le frequenze rilevate sono ben definite, nella terza l'analisi delle frequenze è notevolmente più confusa, mentre si riescono a vedere distintamente le onde sinusoidali del segnale a più bassa frequenza.

Questo significa che operazioni come il *downsampling* (o sottocampionamento del segnale che approfondiremo in seguito), l'*upsampling*, la moltiplicazione per un fattore di scala, non possono incidere sulle dimensioni del cosiddetto *Heisenberg box*; qualunque possibile manipolazione della sequenza porta alla diminuzione della deviazione standard di una grandezza, e all'aumento dell'altra (figura (1.11)).

1.5.1 Suddividere il piano tempo-frequenza

Dopo aver studiato le proprietà di localizzazione di una sequenza individuale nelle due sezioni precedenti, il prossimo passo intuitivamente è quello di considerare insiemi di sequenze. Consideriamo infatti di prendere una sequenza come prototipo e generare tutto l'insieme tramite operazioni base di spostamento nel tempo, in frequenza e moltiplicazioni per uno scalare. Questa semplicità nella descrizione dell'insieme è essenziale in applicazioni, in particolare perché la struttura spesso porta ad algoritmi efficienti per l'analisi e la sintesi.

Definiamo lo *schema reale tempo-frequenza* dell'insieme mostrando su un piano tutti i *rettangoli di Heisenberg* delle diverse sequenze; i diversi rettangoli possono ovviamente risultare sovrapposti o anche completamente separati. Data una sequenza prototipo φ , generiamo una collezione di sequenze utilizzando le tre operazioni seguenti (figura 1.9):

i. *Spostamenti nel tempo* di una quantità $m\,n_0$, con m∈Z, ed $n_0 \in Z^+$:

$$\phi = \{\varphi_n - m\,n_0\}_{m \in Z}; \tag{1.21}$$

ii. *Spostamenti in frequenza (modulazione)* di una quantità $k\,\omega_0$, con k∈ $\{0, 1, \cdots, N-1\}$, ed $\omega_0 \in (-\pi, pi]$:

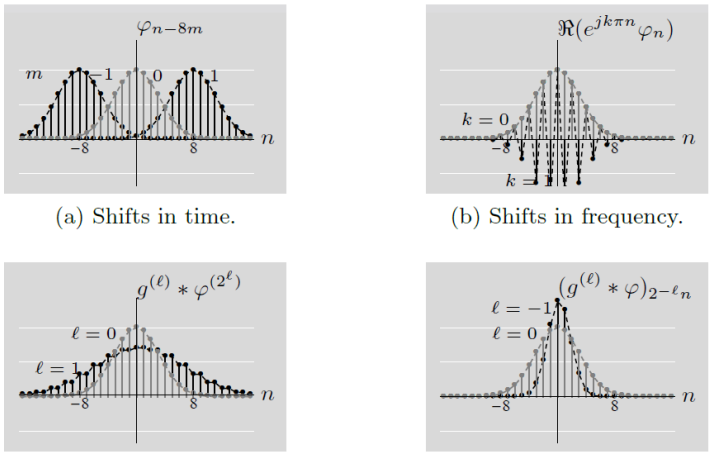
$$\phi = \{e^{jk\omega_0 n}\varphi_n\}_{k \in \{0,1,\cdots,N-1\}}; \tag{1.22}$$

Solitamente si sceglie $\omega_0 = \frac{2\pi}{N}$ per qualche N$\in Z^+$; poiché infatti la trasformata di φ è 2π-periodica, l'insieme finito dei k elencato sopra produce N spostamenti in frequenza tutti distinti, mentre il resto sarebbe identico con periodo 2π.

iii. *Sovracampionamento* (o *Upsampling*) di una quantità N^{l-1} con l$\in Z^+$, N$\in Z^+$, $N > 1$, seguito da un appropriato postfiltraggio,

$$\phi = \{g^{l-1} * \varphi^{N^{l-1}}\}_{l \in Z^+}; \tag{1.23}$$

dove $\varphi^{N^{l-1}}$ rappresenta la sequenza prototipo φ sovracampionata di N^{l-1} , g^{l-1} è un filtro passabasso appropriato che dipende dalla scala con g$^{(}$1)=g e g$^{(}$0) $= \delta$, e N è tipicamente un intero piccolo.



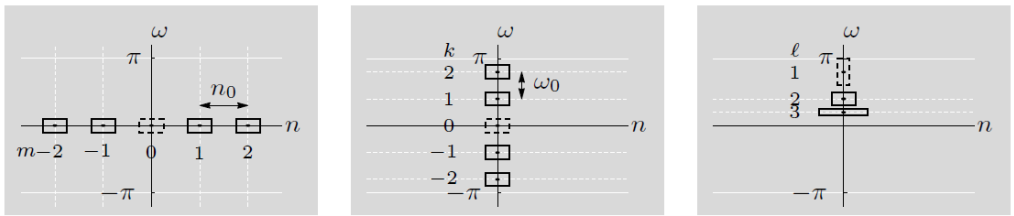
(a) Shifts in time.

(b) Shifts in frequency.

(c) Upsampling followed by filtering.

(d) Downsampling preceded by filtering.

Figura 1.13: Operazioni basi su una sequenza Gaussiana $\varphi = 2^{\frac{1}{4}} e^{-\pi(\frac{n}{8})^2}$. (Illustrato per $n_0 = 8, \omega_0 = \pi, N = 2 e g_n = \frac{\delta_n + \delta_{n-1}}{\sqrt{2}}$



(a) Shifts in time.

(b) Shifts in frequency (modulation).

(c) Upsampling followed by filtering.

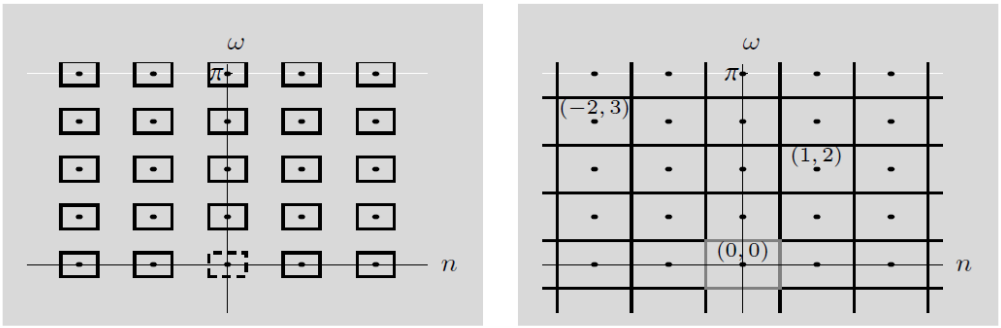
Figura 1.14: Ricoprimento del piano tempo-frequenza risultante dalle operazioni base sulle sequenze. La linea tratteggiata rappresenta il *rettangolo di Heisenberg* della sequenza prototipo.

Come esempio prendiamo due tipi di insiemi capaci di coprire tutto il piano tempo-frequenza, basati su operazioni diverse: lo spostamento nel tempo combinato con la modulazione (o spostamento in frequenza), oppure combinato con la moltiplicazione di uno scalare.

Il primo viene generato da una sequenza prototipo φ, tipicamente passabasso; questo vuol dire che la sequenza contiene tutte le frequenze più basse di un certo valore e nessuna sopra. Dati, quindi, n_0 e N$\in Z^+$, viene generato l'insieme:

$$\varphi_{k,m,n} = e^{jk\omega_0 n}\,\varphi_{n-mn_0}, \qquad k \in \{0,1,\ldots,N-1\}, m \in Z \tag{1.24}$$

con $\omega_0 = \frac{2\pi}{N}$. L'insieme rappresentato sul piano tempo-frequenza è visibile nella figura 1.11 e, come si può notare, i rettangoli sono tutti identici. Le basi della Trasformata di Fourier Locale rispecchiano proprio questo schema, basta ricordare infatti la suddivisione dell'immagine in sottoimmagini 8 x 8, considerando che al posto del tempo dobbiamo sostituire lo spazio e che il segnale non è localizzato in frequenza.

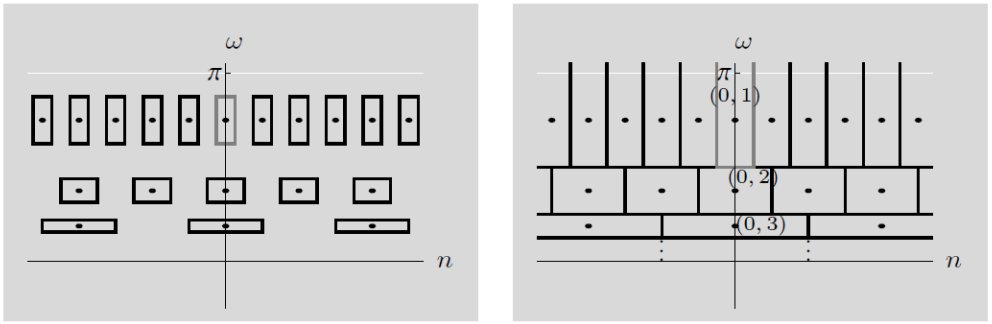


(a) Actual plot. (b) Idealized tiling.

Figura 1.15: Confronto tra il ricoprimento idealizzato e quello reale del piano tempo-frequenza generato dallo spostamento nel tempo e in frequenza.

L'insieme che suscita maggior interesse è, però, quello creato a partire da una sequenza prototipo PHI, tipicamente passaalto che, al contrario di prima, lascia passare le frequenze più alte. Il gruppo di sequenze generano *rettangoli di Heisenberg* di dimensioni differenti perciò, usando spostamenti nel tempo equamente spaziati si ottiene una densità più alta di *rettangoli di Heisenberg* per frequenze basse, e densità più basse per le frequenze alte come si può vedere dalla FIGURA 7.21. La definizione per n_0 e N$\in Z^+$, con N$>$1, risulta:

$$\varphi_{k,m,n} = (g^{l-1} * \varphi^{N^{l-1}})_{n-mN^{l-1}n_0}, \qquad l \in Z^+, \quad m \in Z \tag{1.25}$$



(a) Actual plot. (b) Idealized tiling.

Figura 1.16: Confronto tra il ricoprimento idealizzato e quello reale del piano tempo-frequenza generato dallo spostamento nel tempo e trasformazioni di scala.

Una suddivisione del piano di questo tipo può essere ottenuta proprio con le basi wavelet discrete nel tempo che discuteremo nel prossimo capitolo.

Capitolo 2

Analisi Multirisoluzione e Trasformate Wavelet

Come abbiamo anticipato nel primo capitolo, la Trasformata Wavelet è probabilmente la soluzione più recente ai problemi della Trasformata di Fourier. Quest'ultima è stata, sin dalla fine degli anni '50, la pietra miliare dell'elaborazione di immagini, come abbiamo precedentemente spiegato. Nel 1987 le wavelet sono state presentate per la prima volta come il fondamento di un metodo nuovo ed efficace per l'elaborazione e l'analisi dei segnali: la cosiddetta *Teoria Multirisoluzione*. [9]

La Trasformata Wavelet, diversamente da quella di Fourier, si basa su *piccole onde* ("wavelet", in linguaggio anglosassone), di frequenza variabile e *durata limitata*. In questo modo all'interno di un concerto potremmo analizzare il suono e ricostruire non solo le note che sono state suonate, ma anche il momento in cui questo è avvenuto.

Nell'analisi con le wavelet, l'utilizzo di una finestra completamente modificabile in dimensioni e posizione risolve, infatti, il problema del taglio del segnale. Questo avviene grazie al fatto che la finestra permette di calcolarne lo spettro in ogni posizione e questo processo può essere ripetuto molte volte con una finestra sempre più stretta. Alla fine dell'analisi otterremo in questo modo una collezione di rappresentazioni tempo-frequenza del segnale, ognuna con una specifica risoluzione. Questa collezione genera l'*Analisi Multirisoluzione*. Nel caso delle wavelet, perciò, non parleremo dello spazio tempo-frequenza, ma di quello tempo-scala, dove la scala rappresenta l'opposto della frequenza. Per essere chiari, quando si parla di grande e piccola scala stiamo mostrando rispettivamente l'immagine nella sua interezza, e i suoi dettagli più piccoli. Possiamo, però, capire meglio questo concetto sfruttando proprio i principi dell'analisi multirisoluzione. [1] È molto importante inoltre notare che, da qui in avanti, quando parleremo di risoluzione non ci riferiremo al numero di punti rappresentabili in un'immagine (che si misura in numero di pixel), ma alla densità dei punti che vengono rappresentati, che rappresenta il numero di pixel presenti in uno spazio lineare.

2.1 Analisi Multirisoluzione

Come indica il nome stesso, la teoria multirisoluzione è legata alla rappresentazione e all'analisi di segnali (o immagini) a più risoluzioni con il chiaro obiettivo di individuare le caratteristiche di interesse alla risoluzione in cui queste ultime siano più facilmente individuabili. Per semplificare la trattazione discuteremo inizialmente di segnali che si propagano nel tempo, successivamente applicheremo i risultati anche alle immagini (cioè segnali in 2-D propagati nello spazio).

Quando osserviamo un'immagine, vediamo figure simili e livelli di intensità che si combinano tra loro per formare degli oggetti. Solitamente, infatti, se gli oggetti sono di piccole dimensioni (piccola scala) o a basso contrasto, è più utile un'analisi ad alta risoluzione; se sono grandi è sufficiente una visione d'insieme (larga scala, quindi bassa risoluzione). Se perciò sono presenti oggetti sia grandi che piccoli, può essere vantaggioso studiarli a dimensioni diverse. Per questo motivo è utile riorganizzare le informazioni sull'immagine in un set di dettagli che appaiono a risoluzioni differenti.

In particolare possiamo definire una sequenza di risoluzioni crescenti $(r_j)_{j\in Z}$, in cui i dettagli di un'immagine alla risoluzione r_j sono definiti come la differenza d'informazione tra la sua approssimazione alla risoluzione r_j e la sua approssimazione alla risoluzione minore r_{j-1}. (vedi figura 2.1)

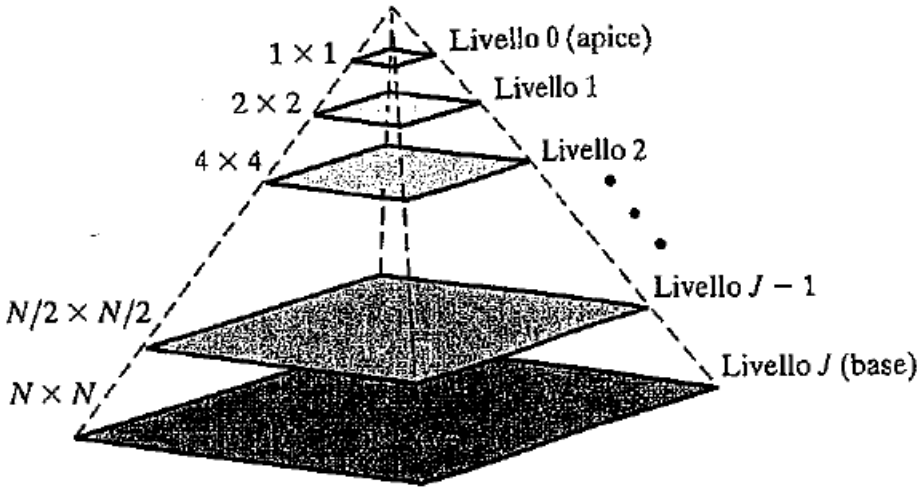


Figura 2.1: Una piramide di immagine. Si può notare che in realtà le scale più alte contengano pochissime informazioni facendo solamente una media dell'immagine. Per questo motivo spesso viene preso in considerazione solo un numero limitato di scale.

Una decomposizione in multirisoluzione ci permette di avere un'interpretazione *invariante in scala* dell'immagine, almeno parzialmente. Infatti la scala di un'immagine varia con la distanza che poniamo tra noi e l'oggetto: avvicinandoci ad esso (o facendo

uno zoom dell'immagine) ingrandiamo la scala, altrimenti la riduciamo. Però, anche se modifichiamo la scala, nella decomposizione in multirisoluzione l'immagine non deve variare, almeno parzialmente: cambieranno i dettagli che si riescono a distinguere ma la trama e i contorni dell'immagine rimangono simili. Infatti una rappresentazione in multirisoluzione può essere *invariante in scala* se la sequenza dei parametri di risoluzione $(r_j)_{j\in Z}$ varia esponenzialmente.

Supponiamo quindi di stabilire un opportuno valore di risoluzione $\alpha \in R$ in modo tale che per qualunque j valga la relazione $r_j = \alpha^j$. Se ci avviciniamo alla scena di α volte, ogni oggetto della scena viene proiettato in un'area α^2 volte più grande. Perciò i dettagli di questa nuova immagine corrispondono ai dettagli dell'immagine precedente alla risoluzione α^{j+1}. Riscalando, quindi, la seconda immagine (che risulta al momento più grande di un fattore α), otteniamo un'immagine i cui dettagli sono traslati lungo l'asse della risoluzione. [9]

In particolare possiamo stabilire il seguente teorema (per la dimostrazione si rimanda all'articolo di Mallat [9]):

Teorema 6 *Sia $(V_{2^j})_{j\in Z}$ un'approssimazione in multirisoluzione dello spazio $L^2(R)$. Allora esiste una e una sola funzione $\phi(x) \in L^2(R)$, chiamata funzione di scala tale che*

$$(\sqrt{2^{-j}}\phi_{2^j}(x - 2^{-j}n)_{n\in Z} \tag{2.1}$$

Risulta una base ortonormale di V_{2^j}, dove $\phi_{2^j}(x) = 2^j\phi(2^j x)$, per qualunque $j \in Z$.

Il teorema mostra che possiamo costruire una base ortonormale per qualunque immagine che fa parte della nostra *piramide multirisoluzione* (cioè il sottospazio di L^2, V_{2^j}), dilatando una funzione $\phi(x)$ con un coefficiente 2^j e traslando la funzione risultante su una griglia il cui intervallo è proporzionale a 2^{-j}.

Questo approccio coincide proprio con quello che avevamo utilizzato per descrivere il ricoprimento dello spazio tempo-frequenza nella figura 1.15. Infatti dilatando la funzione $\phi(x)$ con un coefficiente 2^j, riusciamo a coprire un intervallo di frequenze maggiore. Perciò, grazie al principio di indeterminazione di Heisenberg applicato alle immagini spiegato nel teorema 4, possiamo restringere l'intervallo di tempo (o spazio, nelle immagini) coperto dal coefficiente della trasformata.

Una rappresentazione in multirisoluzione fornisce in questo modo una piramide gerarchica molto utile per l'interpretazione dell'informazione sull'immagine. A differenti risoluzioni, i dettagli di un'immagine generalmente caratterizzano strutture fisiche differenti della scena e questa strategia che passa dalla risoluzione più grezza a quella più fine è molto utile per algoritmi di riconoscimento dei contorni. Alla risoluzione più grezza, infatti questi dettagli corrispondono alle strutture più grandi che forniscono il contesto dell'immagine.

Basandosi su queste caratteristiche peculiari, nelle prossime sezioni andremo a chiarire la maniera in cui definire le wavelet e ottenere l'analisi suddetta.

2.2 La Trasformata Wavelet Continua

L'analisi Wavelet descritta più sopra, porta intuitivamente a definire una *Trasformata Wavelet Continua* o *CWT* molto simile a quella di Fourier basata sulla scomposizione della funzione f(t) rispetto ad una nuova base $\psi_{s,\tau}$ che ricopra tutto lo spazio tempo-scala:

$$\hat{\psi}(s,\tau) = \langle f,\, \psi_{s,\tau} \rangle_{L_2(R)} = \int f(t)\, \frac{1}{\sqrt{s}}\, \psi^* \left(\frac{t-\tau}{s} \right) dt \tag{2.2}$$

Dove * simboleggia il complesso coniugato. Le variabili s e τ, scala e traslazione nel tempo, sono quindi le nuove dimensioni dopo la trasformata wavelet. Le Wavelet che considereremo devono però avere particolari proprietà per permettere l'analisi richiesta.

Definizione 4 *Posto che $\psi \in L^2(R)$ sia una funzione con media zero e norma unitaria, cioè*

$$\int_{-\infty}^{\infty} \psi(t)dt \;=\; 0 \tag{2.3}$$

E

$$||\psi||_{L^2(R)} = 1 \tag{2.4}$$

Chiameremo ψ, Wavelet.

Un esempio di Wavelet coincide con la derivata seconda della Gaussiana, cioè la *Wavelet a Cappello Messicano.* (figura 2.2)

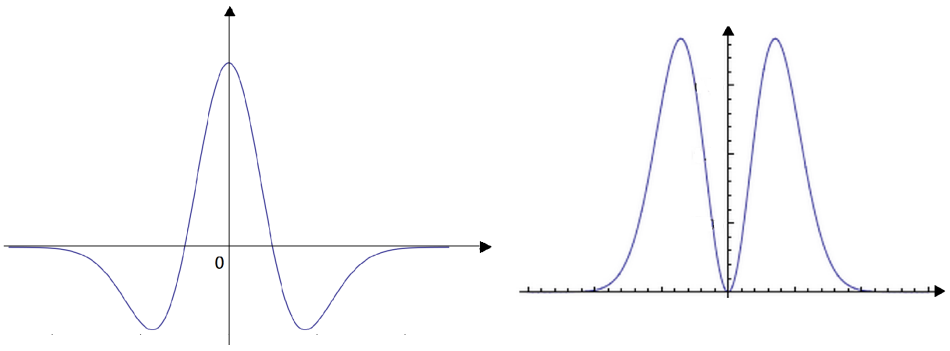


Figura 2.2: La prima immagine raffigura la wavelet con equazione: $\psi(t) = -(\frac{t^2}{\sigma^2} - 1)\, e^{-\frac{t^2}{2\sigma^2}}$. La seconda immagine rappresenta la sua trasformata $\hat{\psi}(\omega) = \omega^2\, e^{-\frac{\sigma^2\omega^2}{2}}$

Quindi, in generale, poiché le wavelet devono avere media nulla, la loro trasformata risulta come la *risposta in frequenza* di un *filtro passabanda.* È richiesta inoltre un'importante proprietà denominata *condizione di ammissibilità*, più restrittiva di quella appena

enunciata, che permette alle funzioni quadrato-integrabili (o $\in L^2(R)$) di essere usate prima per analizzare e poi ricostruire il segnale senza alcuna perdita d'informazione (per la dimostrazione di questo si rimanda alla fonte [3]):

$$C_\psi \ := \int_0^\infty \frac{|\psi\hat{(\omega)}|^2}{|\omega|} d\omega \ < +\infty \tag{2.5}$$

La Wavelet $\psi(t)$ deve quindi avere derivata continua, perché non avere discontinuità garantisce che la sua trasformata decada in modo sufficientemente veloce e C_ψ risulta finito. In tal caso possiamo invertire la trasformata Wavelet tramite la formula:

$$f(t) = \frac{1}{C_\psi} \int_0^\infty \int_{-\infty}^\infty \hat{\psi}(s,\tau) \frac{1}{\sqrt{s}} \psi \left(\frac{t-\tau}{s} \right) d\tau \frac{ds}{s^2} \tag{2.6}$$

La terza ed ultima proprietà richiesta riguarda le *condizioni di regolarità.*

Come possiamo vedere dall'ultima equazione, infatti, la trasformata wavelet di una funzione unidimensionale è 2-dimensionale e la trasformata di una funzione 2-dimensionale, come quelle che approfondiremo nell'ambito delle immagini, è 4-dimensionale. Il prodotto "tempo-larghezza di banda" della trasformata è il quadrato del segnale in input, poiché ogni trasformata della singola wavelet è definita su tutto lo spazio, e per la maggior parte delle applicazioni pratiche questo è da evitare perché per ricostruire ed analizzare la funzione devo fare un integrale quadruplo esteso ogni volta a tutto lo spazio. Per questo motivo cerco di limitare la parte di dominio in cui la funzione si sviluppa in modo da poter evitare di analizzare la parte in cui tende a zero. In particolare impongo qualche condizione sulle funzioni wavelet in modo tale che decrescano abbastanza velocemente al decrescere della scala s, la quale è strettamente legata al tempo (ricordiamo che nell'analisi multirisoluzione alla piccola scala corrisponde un δt della funzione wavelet piccolo).

Poiché la regolarità è un concetto abbastanza complesso, cercheremo di esprimerlo sfruttando quello dei *momenti della funzione.* Se noi infatti espandiamo la trasformata wavelet in serie di Taylor per t$\to$0 fino all'ordine n, e ponendo $\tau = 0$ per comodità, otteniamo:

$$\hat{\psi}(s,0) = \frac{1}{\sqrt{s}} \left[\sum_{p=0}^{n} f^{(p)}(0) \int \frac{t^p}{p!} \psi \left(\frac{t}{s} \right) dt + O(n+1) \right] \tag{2.7}$$

Dove $f^{(p)}$ sta per la derivata p-esima di f e O(n+1) sta per il resto dell'espansione. Se ora definiamo i *momenti* delle wavelet con M_p, con

$$M_p = \int t^p \, \psi(t) dt \tag{2.8}$$

Allora possiamo riscrivere lo sviluppo finito come

$$\hat{\psi}(s,0) = \frac{1}{\sqrt{s}} \left[f(0) M_0 s + \frac{f^{(1)}(0)}{1!} M_1 s^2 + \frac{f^{(2)}(0)}{2!} M_2 s^3 + \cdots + \frac{f^{(n)}(0)}{n!} M_n s^{n+1} + O(s^{n+2}) \right] \tag{2.9}$$

Dalla condizione di ammissibilità noi possiamo già sapere che il momento 0 è nullo,$M_0 = 0$, perciò dobbiamo riuscire a trovare una wavelet che abbia tutti gli altri momenti nulli fino all'ordine N in modo tale che i coefficienti delle trasformate wavelet $\hat{\psi}(s,\tau)$ decadano come s^{N+2} per un segnale continuo f(t) e potremo dire che l'*ordine di approssimazione* è N.(Un ulteriore approfondimento sulle condizioni di regolarità si può trovare nel libro di Daubechies [5]) È interessante notare che il motivo per cui originariamente le wavelet erano chiamate "ondelette" (cioè "ondine" in francese)risiede nella loro forma che, a causa dell'annullamento dei momenti della funzione, va a zero rapidamente (vedi figura 2.3. Ora che abbiamo stabilito le basi teoriche e le proprietà necessarie alla nostra analisi multirisoluzione tramite trasformate wavelet, possiamo cercare di renderle più pratiche in modo tale da sfruttarne tutte le potenzialità.

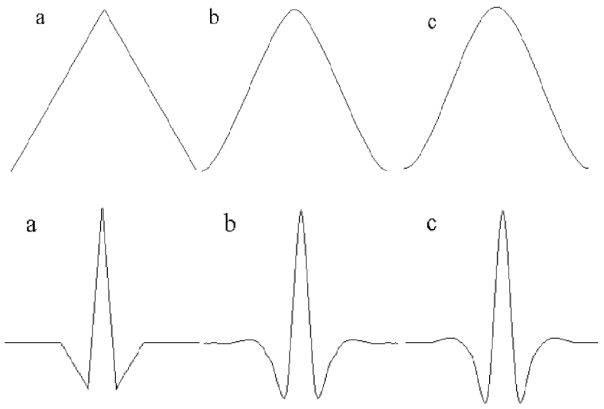


Figura 2.3: Alcune funzioni di scala (sopra) e wavelets (sotto) della famiglia biortogonale di Deslauriers-Dubuc con coefficienti (m,n)= (2,2), (4,2), (6,2), dove m rappresenta il numero di momenti nulli della wavelet di analisi (cioè quella che scompone il segnale), e n rappresenta il numero di momenti nulli della wavelet di sintesi (cioè quella che ricostruisce il segnale). Da notare che il numero crescente di momenti nulli spesso può rendere la wavelet più liscia e regolare, ma vedremo in seguito che non è sempre vero.

2.3 Wavelet Discrete

La trasformata wavelet descritta nel paragrafo precedente ha ancora tre proprietà che rendono la sua formulazione difficile da applicare. La prima è la ridondanza della CWT. Nell'equazione 2.2 la trasformata è calcolata utilizzando funzioni che traslo nel tempo e in scala in modo continuo. Questo implica che inevitabilmente queste funzioni vanno a sovrapporsi e non potranno mai essere ortogonali tra loro e per questo i coefficienti della trasformata saranno altamente ridondanti. Abbiamo già spiegato quali sono le conseguenze di avere un sistema ridondante nel paragrafo 1.2 e in ultima analisi si possono

ricondurre tutte ad una compressione meno efficiente, e solo in piccola misura si rivelano utili per l'eliminazione del rumore e degli errori.

Una volta eliminata la ridondanza, rimangono comunque da considerare un numero infinito di wavelets nella CWT e perciò è richiesto di ridurre questo numero ad una quantità finita.

Il terzo problema riguarda il fatto che la maggior parte delle funzioni non hanno soluzioni analitiche e possono essere calcolate solo tramite metodi numerici. È infatti l'esistenza di algoritmi più veloci che ha permesso il diffondersi delle Wavelet. Nei prossimi paragrafi affronteremo uno ad uno i tre problemi cercando le possibili soluzioni.

2.3.1 Ridondanza

Come detto la CWT mappa un segnale 1-dimensionale in una rappresentazione congiunta tempo-scala altamente ridondante e il prodotto *tempo-larghezza di banda* risulta essere il quadrato del segnale e questo rende la trasformata poco efficiente. Per risolvere questo problema sono quindi state introdotte le *Wavelet Discrete*.

La differenza tra queste ultime e quelle utilizzate precedentemente consiste nel fatto che la *Wavelet Discrete* non possono essere traslate o scalate in modo continuo, ma solo attraverso passi discreti. Per questo motivo ridefiniamo le wavelet come:

$$\psi_{j,k}(t) = \frac{1}{\sqrt{s_0^j}} \psi \left(\frac{t - k\tau_0 s_0^j}{s_0^j} \right) \tag{2.10}$$

Dove j e k sono interi, $s_0 > 1$ rappresenta una dilatazione della funzione a passo costante e il fattore di traslazione τ_0 dipende da s_0. La funzione risulta quindi continua (almeno a pezzi), ma discretizzando la wavelet abbiamo discretizzato il modo in cui ricopriamo lo *spazio tempo-scala* che è ora campionato ad intervalli discreti. In particolare, solitamente si sceglie $s_0 = 2$ in modo tale che il campionamento sull'asse della frequenza corrisponde ad un *campionamento diadico*. Questa infatti è una scelta molto naturale e conveniente per i computer (che non a caso si basano su codici binari), e ricalca da vicino la conformazione dell'orecchio[1] e quindi la musica per esempio. Scegliamo infine $\tau_0 = 1$ in modo tale da avere un campionamento diadico anche sull'asse temporale. Tramite

[1]Il nostro orecchio utilizza proprio una trasformata wavelet quando analizza il suono, almeno nel primo stadio. Le oscillazioni dell'ampiezza di pressione sono trasmesse dal *tamburo* alla *membrana basilare*, la quale si estende per tutta la lunghezza della *coclea*. La coclea, conosciuta anche come chiocciola, si arrotola come una spirale nella parte più interna del nostro orecchio. Se immaginiamo quindi di srotolarla lungo una retta, lo farà anche la membrana basilare e potremo introdurre una coordinata y lungo il segmento. Esperimenti e simulazioni numeriche mostrano che un'onda di pressione, cioè una nota pura $f_\omega(t) = e^{i\omega t}$, porta ad un'eccitazione di risposta lungo la membrana basilare che ha la stessa frequenza nel tempo, ma con un inviluppo in y, $F_\omega(t, y) = e^{i\omega t}\, \phi_\omega(y)$ e questo ricorda già le wavelet poiché l'inviluppo dipende proprio dalla frequenza . Poiché poi in prima approssimazione la dipendenza di ϕ da ω corrisponde ad uno spostamento di $\log\omega$, e poiché per una funzione generale

queste nuove funzioni, la trasformata di un segnale continuo sarà una serie di coefficienti wavelet e non più una funzione continua, e sarà chiamata *decomposizione in serie di wavelet*.

Un problema fondamentale da analizzare risulta però il *problema della ricostruzione*, cioè verificare che sia possibile ricostruire il segnale in modo preciso dopo aver eseguito la trasformata. Si può infatti provare [5] che una condizione necessaria e sufficiente per una ricostruzione stabile è che l'energia dei coefficienti wavelet deve essere compresa tra due valori positivi:

$$A||f||^2 \leq \sum_{j,k} |\langle f, \psi_{j,k} \rangle|^2 \leq B||f||^2 \tag{2.11}$$

Dove $||f||^2$ rappresenta l'energia di f(t), A>0, B<∞ e A,B sono indipendenti da f(t) ma solo dalle funzioni wavelet. Quando la relazione precedente è soddisfatta, diremo che la famiglia delle funzioni $\psi_{j,k}(t)$ forma una base di L^2, inteso come insieme di funzioni linearmente indipendenti e non obbligatoriamente ortogonali tra loro. Quest'ultima condizione è soddisfatta solo quando A=B, e in quel caso diremo che la base è piccola e le wavelet discrete si comportano come una base ortonormale. VIceversa quando $A \neq B$ una ricostruzione esatta è ancora possibile ma a spese di una base duale, nel quale le wavelet di decomposizione sono differenti da quelle di ricostruzione. Questo comportamento, perciò, per la maggior parte degli scopi non è auspicabile. Lasciamo ora da parte le basi poiché, diversamente dalla serie di Fourier, l'argomento è molto ampio in quanto l'analisi wavelet lascia grande libertà nella scelta, e continuiamo a rimuovere tutte le ridondanze.

L'ultimo passo per avere un sistema di wavelet efficienti e non ridondanti è verificare la loro ortonormalità, e questo può avvenire tramite una scelta particolare della *wavelet madre* dalla quale, tramite dilatazioni e traslazioni, otteniamo tutte le altre:

$$\int \psi_{j,k}(t)\psi^*_{m,n}(t)dt = \begin{cases} 1 & se\ j = m\ e\ k = n \\ 0 & altrimenti \end{cases} \tag{2.12}$$

Tramite queste nuove funzioni di base wavelet discrete e ortogonali, possiamo riscrivere la trasformata inversa e ricostruire il segnale:

$$f(t) = \sum_{j,k} \hat{\psi}(j,k)\, \psi_{j,k}(t) \tag{2.13}$$

Oltre al ridurre la sensibilità ai rumori della rappresentazione del segnale, la ridondanza migliora anche l'*invarianza rispetto alla traslazione*. Questo infatti rappresenta uno

di eccitazione f abbiamo $f(t) = \frac{1}{\sqrt{2\,\pi}} \int \hat{f}(\omega)\, e^{i\omega t} d\omega$, basterà introdurre un cambio di parametrizzazione $\psi(e^{-x}) = (2\pi)^{-1/2}\, \phi(x)$ e $G(a,t) = F(t, \log a)$ per ottenere la corrispondente sovrapposizione di "funzioni elementari di risposta", cioè la funzione di risposta $G(a,t) = \int f(t')\psi(a(t-t'))dt'$ che coincide proprio con la trasformata wavelet. Per un ulteriore approfondimento e per la dimostrazione completa si rimanda a [5].

svantaggio delle wavelet discrete la cui trasformata di un segnale differisce da quella del segnale traslato nel tempo in modo significativo, e non sono semplicemente versioni traslate l'una dell'altra.

2.3.2 Un filtro passa-banda

Rimossa la ridondanza, rimangono da trattare ancora due problemi. Il primo è quello che riguarda l'illimitatezza della base: abbiamo infatti ancora bisogno di un numero infinito di variazioni di scala e traslazioni al fine di calcolare la trasformata wavelet.

Il modo più intuitivo di porre rimedio a questo problema è semplicemente quello di limitare il numero delle wavelet considerate nel processo. Ma questo suscita automaticamente la domanda sulla qualità della trasformata e sulle possibili perdite derivanti da essa. È possibile ricavarne ancora un risultato utile? Ovviamente i problemi riguardano le scale da analizzare e come ottenere un limite inferiore. Infatti per quanto riguarda le traslazioni nel tempo, abbiamo sempre a che fare con segnali con una durata limitata e questo ci fornisce automaticamente un limite superiore.

Per risolvere questo problema proviamo a studiare la trasformata da un altro punto di vista: analizziamo la trasformata come un *filtro passabanda*.

Dall'analisi di Fourier sappiamo che la compressione nel tempo equivale ad aumentare lo spettro della trasformata e a traslarlo:

$$F\{f(at)\} = \frac{1}{|a|} F\left(\frac{\omega}{a}\right) \tag{2.14}$$

Questo significa che possiamo coprire lo spettro finito del nostro segnale nel dominio del tempo tramite le wavelet traslate in modo sufficientemente preciso per la ricostruzione. Per ottenere una buona copertura di tutto lo spettro del segnale, come possiamo ricordare anche dalla figura 1.15, gli spettri coperti dalle singole wavelet dovrebbero toccarsi:

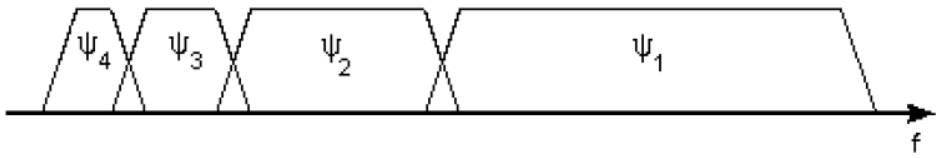


Figura 2.4: Rappresentazione della copertura dello spettro del segnale tramite variazioni di scala della *wavelet madre* nel dominio del tempo.

Riassumendo, se una wavelet può essere vista come un filtro passabanda, allora una serie di wavelet dilatate può essere visto come un *banco di filtri* passabanda.

2.3.3 La funzione di scala

Se quindi in generale riusciamo a coprire lo spettro del segnale a partire dalle alte frequenze, rimane la difficoltà di coprire le frequenze basse dove il banco dei filtri risulta divergente. Poiché infatti ogni volta che ingrandiamo la wavelet nel dominio del tempo di un fattore due la larghezza di banda è dimezzata, riusciamo a coprire solo metà dello spettro rimanente e per questo dobbiamo iterare questo processo un numero infinito di volte ottenendo una quantità ancora illimitata di coefficienti delle funzioni wavelet. La soluzione a questo problema è semplicemente quella di non cercare di coprire lo spettro fino a zero tramite gli spettri delle wavelet, ma di usare un altro metodo: la cosiddetta *funzione di scala*, la quale fu introdotta da Mallat [9] e che consiste in uno spettro passabasso. A causa di questo è denominata anche *filtro di media*. Se analizziamo la funzione di scala come spettro passabasso, allora possiamo decomporla in una somma infinita di componenti wavelet fino alla scala j:

$$\varphi(t) = \sum_{j,k} \hat{\psi}(j,k)\psi_{j,k}(t) \tag{2.15}$$

In questo modo abbiamo ridotto la parte in cui avevo componenti infinite, ad una sola funzione (filtro) che copre le frequenze più basse, cioè le parti costanti del segnale, mentre tutto il resto viene analizzato tramite le proiezioni rispetto alle funzioni di base wavelet (figura 2.5).

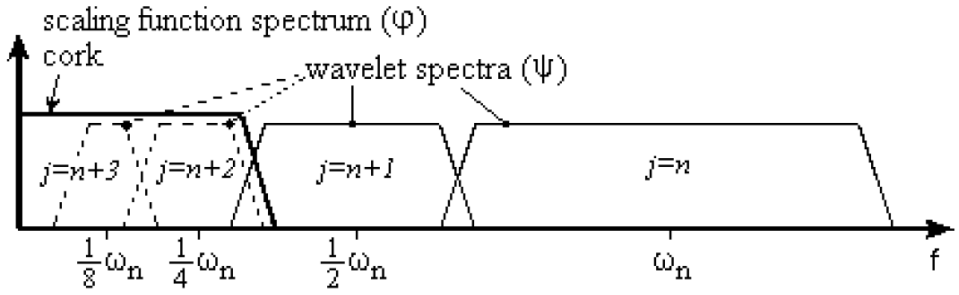


Figura 2.5: Un insieme infinito di wavelet può essere coperto grazie ad una *funzione di scala*.

Ovviamente questo metodo comporta una perdita d'informazioni e perciò dev'essere utilizzato in modo appropriato. In realtà dal punto di vista della rappresentazione del segnale, non perderemo alcuna informazione poiché sarà ancora possibile ricostruire il segnale originale, ma dal punto di vista delle wavelet stiamo perdendo informazioni significative, perciò dovremo ridurre lo spettro coperto dalla funzione di scala al minimo indispensabile.

La natura passabasso dello spettro della funzione di scala ci permette di stabilire una qualche sorta di condizione di ammissibilità:

$$\int \varphi(t)dt = 1 \tag{2.16}$$

e questo giustamente mostra che il momento 0 della funzione scalare non può annullarsi e la sua media non sarà nulla[2].

2.3.4 Il quadro matematico

Per il momento abbiamo descritto le intuizioni che ci hanno portato a definire le trasformate wavelet e le loro principali caratteristiche. Per poter proseguire con l'analisi delle wavelet, la sua ottimizzazione e le possibili applicazioni, però, abbiamo bisogno ora di inquadrarle in modo più rigoroso in un quadro matematico coerente. Ridefiniamo quindi le diverse parti dell'analisi a partire dalla funzione di scala

Funzione di Scala

Possiamo applicare traslazioni e variazioni di scala anche alle funzioni di scala generando l'insieme $\{\varphi_{j,k}(x)\}$, dove

$$\varphi_{j,k}(x) = 2^{j/2}\varphi(2^j x - k) \tag{2.17}$$

Per tutti i valori $j, k \in Z$ e $\varphi(x) \in L^2(R)$. In questo caso ovviamente k determina la posizione, e j l'ampiezza della funzione.

Se consideriamo ora un solo valore j, diciamo j=j_0, l'insieme che ne risulta $\{\varphi_{j_0,k}(x)\}$ è un sottoinsieme di $\{\phi_{j,k}(x)\}$, che a sua volta rappresenta un sottospazio di $L^2(R)$:

$$V_{j_0} = \overline{Span_k \{\varphi_{j_0,k}(x)\}} \tag{2.18}$$

Più in generale potremo denotare il sottospazio su k per ogni valore j come

$$V_j = \overline{Span_k \{\varphi_{j,k}(x)\}} \tag{2.19}$$

Aumentando j, aumentano le dimensioni di V_j, cosa che permette alle funzioni con variazioni più piccole o con dettagli più definiti di essere inclusi nel sottospazio. Questa è una conseguenza del fatto che, mentre j aumenta, i valori $\varphi_{j,k}(x)$ che sono utilizzati per rappresentare le funzioni del sottospazio diventano più strette e separate da piccole variazioni delle x.

[2] Del resto la funzione costante si ottiene solamente per frequenza $\omega = 0$ e questo significa che la scala $s \to \infty$, ed è proprio la parte coperta dalla funzione di scala

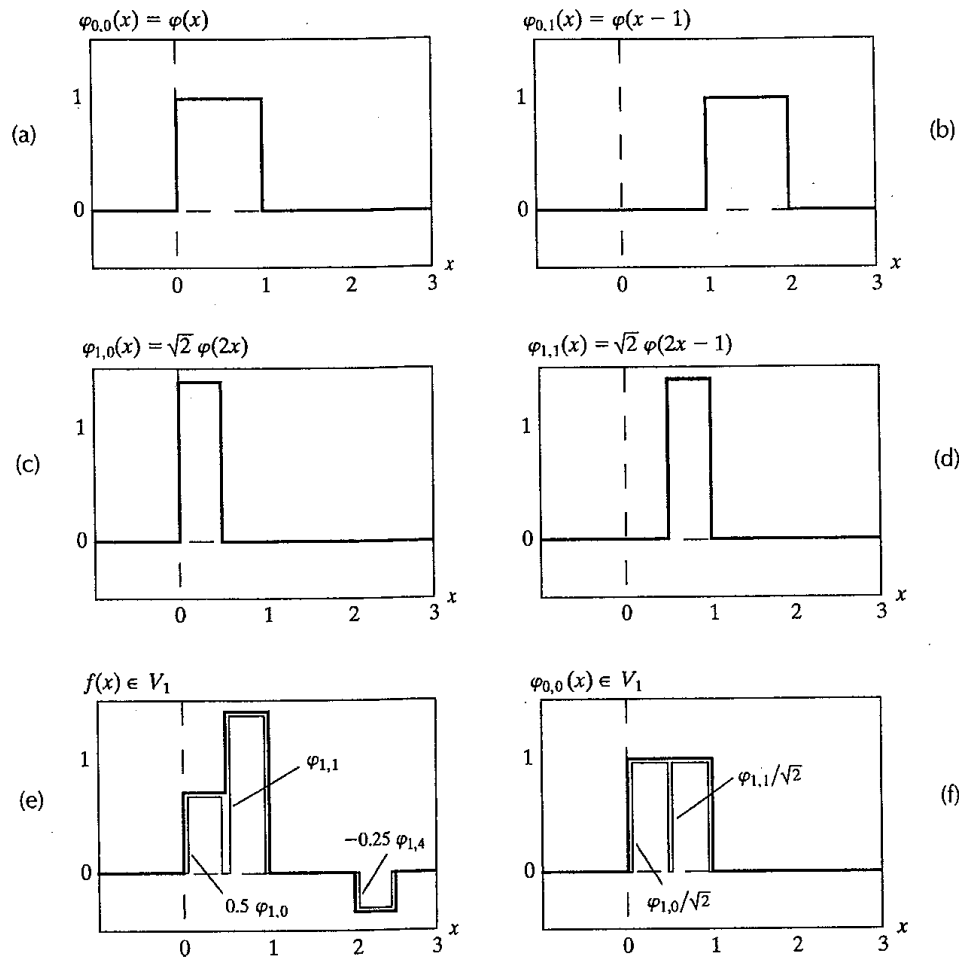


Figura 2.6: Funzioni di scala di Haar. Nelle prime quattro figure sono rappresentate le basi di Haar, la cui equazione è scritta sopra alla figura corrispondente. L'immagine (e) rappresenta la scomposizione di una funzione appartenente a V_1 che risulta $f(x) = 0,5\varphi_{1,0}(x) + \varphi_{1,1}(x) - 0,25\varphi_{1,4}(x)$. La figura (f) illustra la decomposizione di $\varphi_{0,0}(x)$ come somma delle funzioni di base dell'insieme V_1: $\varphi_{0,k}(x) = \frac{1}{\sqrt{2}}\varphi_{1,2k}(x) + \varphi_{1,2k+1}(x)$, dove k in questo caso è 0. Si può notare dalle ultime due immagini, quindi, che V_0 è sottinsieme di V_1 ma non sono affatto coincidenti, poiché la funzione f(x) $\in V_1$ non è rappresentabile tramite le funzioni di base di V_0 poiché sono troppo approssimate per rappresentarla. Si richiedono quindi funzioni a più alta risoluzione come quelle in V_1.

In generale una funzione di scala deve rispettare quattro fondamentali requisiti dell'analisi multirisoluzione (per ulteriori informazioni vedere [9]):

1. *La funzione di scala è ortogonale rispetto alle sue traslazioni intere.* Si dice quindi che la funzione di scala di Haar rappresentata in figura 2.6 ha un *supporto* compatto,

poiché vale 0 al di fuori di un intervallo finito detto *supporto.*

2. *I sottospazi ricoperti dalla funzione di scaling alla scala più bassa vengono annidati all'interno di quelli distribuiti sulle scale più alte.* (vedi figura 2.7)

3. *La sola funzione comune a tutti i valori V_j è f(x)=0.* Risulta infatti che $V_{-\infty} = \{0\}$.

4. *Ogni funzione può essere rappresentata con precisione arbitraria.* Nonostante non sia possibile espandere una particolare funzione f(x) ad una risoluzione arbitrariamente ampia, tutte le funzioni misurabili $\in L^2(R)$ possono essere rappresentate tramite funzioni di scala per j$\to \infty$. Cioè $V_\infty = \{L^2(R)\}$.

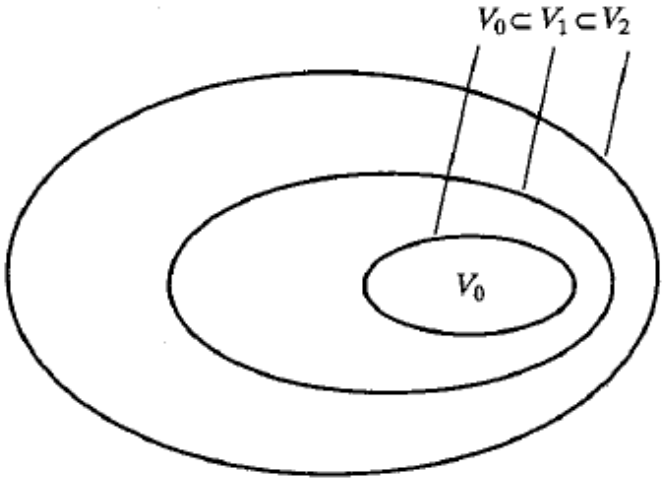


Figura 2.7: Gli spazi di funzione ricoperti da una funzione di scaling. Troviamo che sono annidati l'uno dentro l'altro: $V_{-\infty} \subset \cdots \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \cdots \subset V_\infty$.

Con questi requisiti, come abbiamo visto nella figura 2.6 le funzioni di base del sottospazio V_j possono essere espresse come una somma pesata delle funzioni di base del sottospazio V_{j+1} come:

$$\varphi(t) = \sum_n \alpha_n \varphi_{j+1,n}(x) \tag{2.20}$$

Che con qualche sostituzione e l'utilizzo della definizione di funzione di scala precedente corrisponde a

$$\varphi_{j,k}(t) = \sum_n h_\varphi(n) 2^{\frac{j+1}{2}} \varphi(2^{j+1}x - n) \tag{2.21}$$

Dato che quando scriviamo $\varphi(x)$ intendiamo sempre la *funzione di scala madre* da cui si originano tutte le altre, possiamo sostituirla con $\varphi_{0,0}(x)$ otteniamo:

$$\varphi(t) = \sum_n h_\varphi(n) \sqrt{2} \varphi(2x - n) \tag{2.22}$$

I coefficienti $h_\varphi(n)$ in quest'equazione ricorsiva vengono detti *coefficienti della funzione di scala*, mentre h_φ viene detto *vettore di scaling*. L'ultima equazione è conosciuta come *equazione di rifinimento* o *di dilatazione*. Essa fondamentalmente afferma in generale quello che intuitivamente avevamo già accennato: le funzioni di base di ogni sottospazio possono essere costruite da copie di sé stesse a risoluzione doppia.

Perciò, mentre prima avevo solo delle relazioni di inclusione tra i vari insiemi, ora ho definito in modo esplicito e preciso la relazione che intercorre tra le funzioni di base che generano quegli insiemi. Similmente possiamo ora definire e studiare le relazioni esistenti tra gli insiemi delle funzioni wavelet.

Funzioni Wavelet

Sappiamo quindi che i sottospazi V_j sono sempre inclusi in quelli V_{j+1} ma mai coincidenti. Data una funzione di scala, vogliamo perciò identificare una base che ricopre il sottospazio ottenuto dalla differenza tra i due. Definiamo l'insieme $\{\psi_{j,k}(x)\}$, dove

$$\psi_{j,k}(x) = 2^{j/2}\psi(2^j x \breve{} k) \tag{2.23}$$

Per tutti i valori k $\in$ Z che coprono gli spazi W_j nella figura 2.8 .

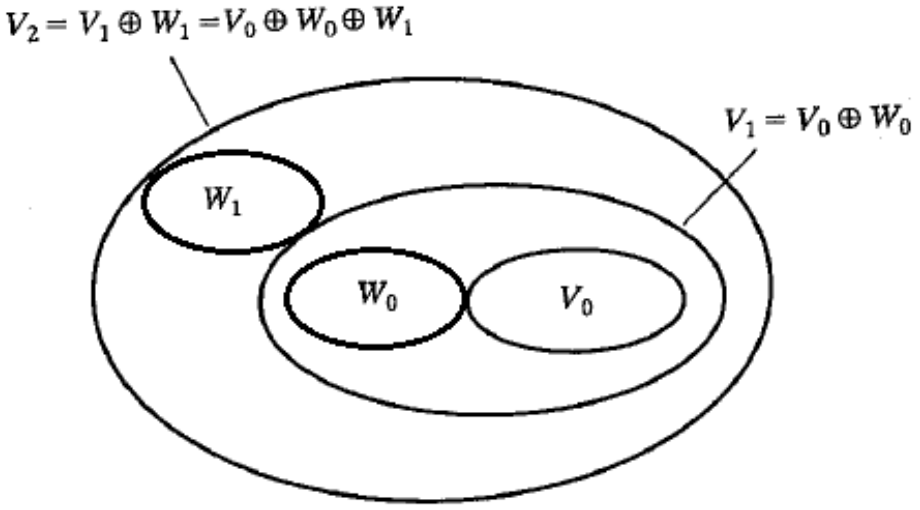


Figura 2.8: La relazione tra gli spazi di funzione wavelet e di scala. È importante notare che le somme sono tutte dirette perché l'intersezione tra gli insiemi è rigorosamente nulla (approfondiremo l'importanza di questo fatto quando parleremo di filtri in quadratura nel capitolo 3).

Dalla figura precedente possiamo notare che i sottospazi delle funzioni di scala e wavelet sono legati da $V_{j+1} = V_j \oplus W_j$. Il complemento ortogonale di V_j in V_{j+1} è quindi

W_j e tutti i membri di W_j sono ortogonali a V_j:

$$\langle \varphi_{j,k}(x), \psi_{j,l}(x) \rangle = 0 \tag{2.24}$$

Per tutti i valori j,k,l $\in$ Z.

Possiamo a questo punto esprimere lo spazio di tutte le funzioni misurabili e integrabili al quadrato come:

$$L^2(R) = V_0 \oplus W_0 \oplus W_1 \oplus \dots \tag{2.25}$$

O in generale considerando una scala iniziale arbitraria j_0:

$$L^2(R) = V_{j_0} \oplus W_{j_0} \oplus W_{j_0+1} \oplus \dots \tag{2.26}$$

O addirittura possiamo eliminare la funzione di scala e rappresentare una funzione in termini di sole wavelet:

$$L^2(R) = \dots \oplus W_2 \oplus W_1 \oplus W_0 \oplus W_1 \oplus W_2 \oplus \dots \tag{2.27}$$

Dato che gli spazi wavelet W_j rappresentano una parte degli spazi V_{j+1} ricoperti dalle funzioni di scala φ_j a più alta risoluzione, ogni funzione wavelet può essere espressa come una somma pesata delle funzioni di scala traslate a risoluzione doppia, cioè:

$$\psi(x) = \sum_n h_\varphi(n) \sqrt{2} \varphi(2x - n) \tag{2.28}$$

Dove $h_\psi(n)$ sono detti *coefficienti della funzione wavelet* e h_ψ è il *vettore wavelet*. Chiariremo però in modo più preciso all'interno del paragrafo della Trasformata Wavelet Veloce che cosa siano e come si possano ottenere.

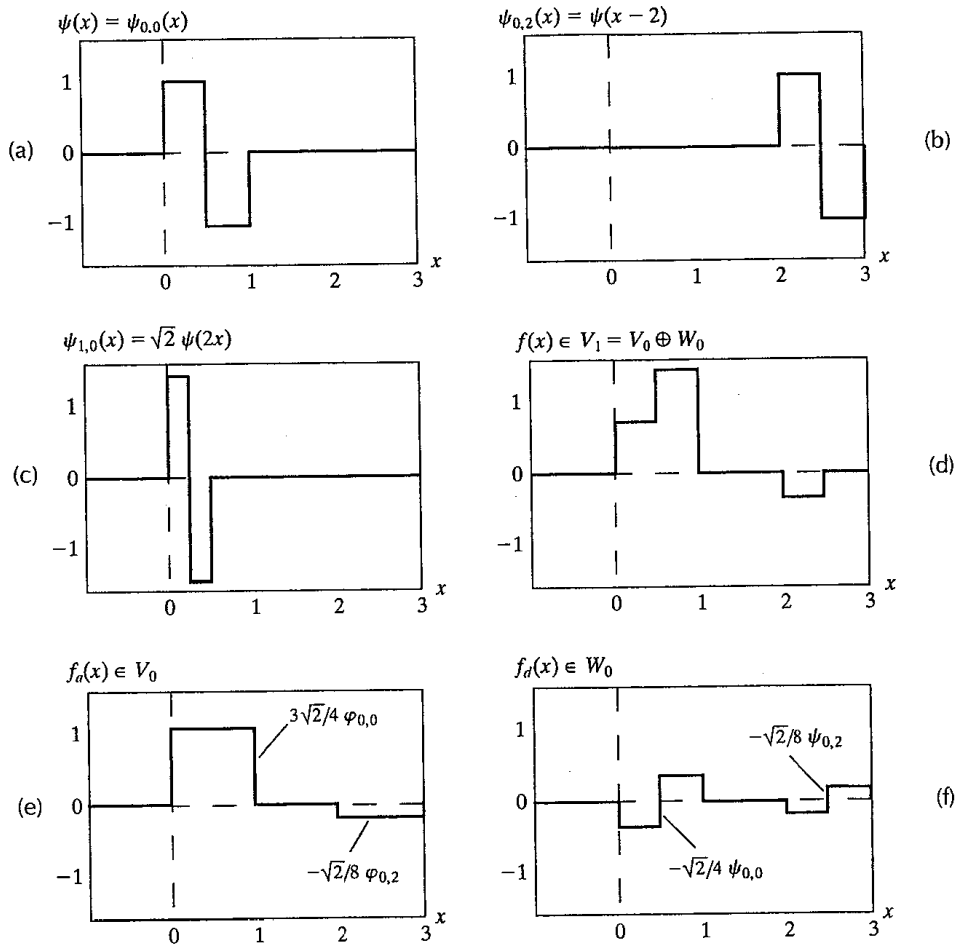
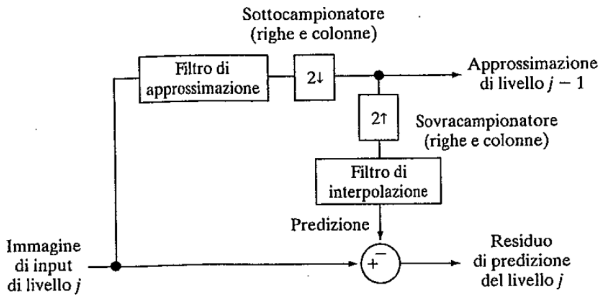


Figura 2.9: Funzioni Wavelet di Haar in W_0 e W_1. L'immagine (d) rappresenta la funzione che risulta somma della funzione $f_a(x)$ in (e), in cui vengono catturate le frequenze più basse, e che per questo assume il valore medio di f(x) in ciascun intervallo intero, e della funzione $f_d(x)$, in cui vengono catturate le frequenze più alte.

Concludiamo questa sezione mostrando un'immagine che raffigura l'idea di analisi multirisoluzione tramite wavelet e funzione di scala. Possiamo notare infatti dalla figura 2.10 che, partendo dalla prima immagine in alto a sinistra (a), abbiamo eseguito un sottocampionamento ottenendo in questo modo una nuova immagine con le frequenze più basse. L'altra piramide di immagini costituisce una *piramide residua di predizione*, cioè rappresenta la differenza che esiste tra l'immagine sopra e quella sottocampionata. In particolare lo schema seguito per la generazione delle due piramidi è il seguente:



Questo schema aiuta a collegare le piramidi di immagini alla scomposizione in sottospazi eseguita precedentemente. Si può notare infatti che la piramide (a) coincide proprio con i sottospazi $\{V_j\}$ generati dalle funzioni di scala, mentre quella sotto (b) coincide con quelli $\{W_j\}$, generati dalle wavelet. Possiamo quindi dedurre che la semplice addizione dell'ultima immagine con tutta la piramide (b) (opportunamente riscalata), costituisce la scomposizione dell'immagine originale in (a).

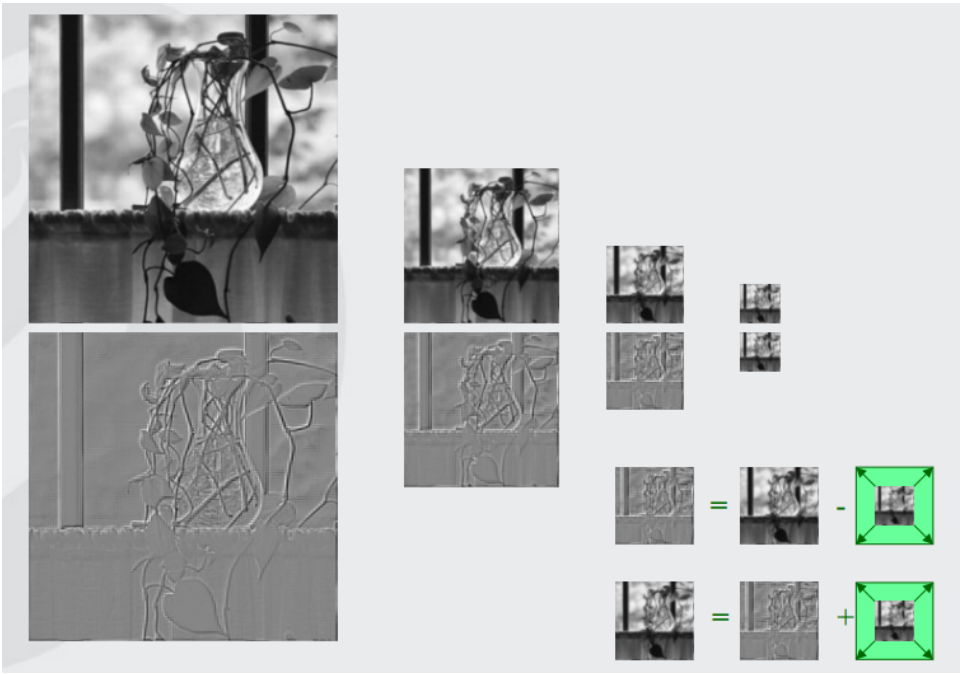


Figura 2.10: Due piramidi di immagini e i loro istogrammi: la piramide di approssimazione, detta piramide gaussiana, e la piramide residua di predizione, detta piramide laplaciana.(Ulteriori informazioni sulla definizione delle due piramidi e sulla loro generazione sono reperibili in [2])

2.4 Trasformate Wavelet in una dimensione

Siamo finalmente in grado di definire formalmente alcune trasformate wavelet strettamente collegate: l' *espansione in serie di wavelet*, la *trasformata wavelet discreta* e la *trasformata wavelet continua*.

Espansione in serie di Wavelet

Cominciamo definendo l'espansione in serie di wavelet della funzione f(x)$\in L^2(R)$ relativa alle wavelet $\psi(x)$ e alla funzione di scala $\phi(x)$. Come abbiamo visto nella sezione precedente, f(x) può essere rappresentata da un'espansione della funzione di scala nel sottospazio V_{j_0} e da un certo numero di basi della funzione wavelet nei sottospazi W_{j_0}, W_{j_0+1}, ... :

$$f(x) = \sum_k c_{j_0}(k)\,\varphi_{j_0,k}(x) + \sum_{j=j_0}^{\infty}\sum_k d_j(k)\,\psi_{j,k}(x) \tag{2.29}$$

Dove j_0 è una scala iniziale arbitraria. I valori c_{j_0} sono detti *coefficienti di approssimazione* o *di scala*, mentre i valori $d_j(k)$ sono detti *coefficienti di dettaglio* o *wavelet*. Infatti, una volta analizzata l'immagine alla scala j_0, gli (eventuali) ulteriori dettagli verranno scomposti in una sommatoria di wavelet, cioè attraverso funzioni con risoluzione più fine.

Se le funzioni di base sono ortonormali tra loro, o comunque formano un textitframe [3]"stretto", allora potremo scrivere i coefficienti di base come:

$$c_{j_0}(k) = \langle f(x), \varphi_{j_0,k}(x)\rangle = \int f(x)\,\varphi_{j_0,k}(x)dx \tag{2.30}$$

e

$$d_j(k) = \langle f(x), \psi_{j,k}(x)\rangle = \int f(x)\,\psi_{j,k}(x)dx \tag{2.31}$$

Sfruttando la definizione di prodotto scalare e ricordando che la funzione f(x) in analisi è reale, perciò $f^*(x) = f(x)$.

2.4.1 Trasformata Wavelet Discreta

Arriviamo finalmente alla definizione della trasformata wavelet che utilizzeremo nell'analisi delle immagini. Poiché infatti la funzione da scomporre è discreta dovremo utilizzare delle funzioni di base discrete, che sono generate dal campionamento delle funzioni continue.

[3]Un *frame* è la generalizzazione del concetto di base poiché è un insieme di vettori linearmente dipendenti che quindi non sono ortonormali tra loro, ma che allo stesso tempo generano tutto lo spazio V

Perciò, se ad esempio avremo un segnale campionato ad intervalli $\triangle x$ fissi come $f(n) = f(x_0 + n\triangle x)$ per alcuni valori n=0,1,...,M-1 , i coefficienti di espansione in serie wavelet risultano:

$$W_\varphi(j_0, k) = \frac{1}{\sqrt{M}} \sum_n f(n)\varphi_{j_0,k}(n) \tag{2.32}$$

$$W_\psi(j, k) = \frac{1}{\sqrt{M}} \sum_n f(n)\psi_{j,k}(n) \qquad perj \geq j_0 \tag{2.33}$$

La Trasformata Wavelet Discreta inversa risulta invece:

$$f(n) =) = \frac{1}{\sqrt{M}} \sum_k W_\varphi(j_0, k)\, \varphi_{j_0,k}(n) + \frac{1}{\sqrt{M}} \sum_{j=j_0}^{\infty} \sum_k W_\psi(j, k)\, \psi_{j,k}(n) \tag{2.34}$$

Solitamente, si pone $j_0 = 0$ e si sceglie M come una potenza di 2 perché, come abbiamo detto, un *campionamento diadico* dello spazio tempo-frequenza è più adatto all'architettura binaria dei calcolatori, all'orecchio umano e alle altre applicazioni. Suddividiamo in questo modo il piano tempo-frequenza secondo le richieste dell'analisi multirisoluzione e avremo $n = 0, 1, 2, \ldots, M-1, \quad j = 0, 1, 2, \ldots, J-1, \quad e\ k = 0, 1, 2, \ldots, 2^j - 1$. Un esempio molto importante di funzioni di base $\varphi_{j_0,k}(n)$ e $\psi_{j,k}(n)$, anche se continue, sono raffigurate nella figura 2.11. Esse sono chiamate basi di Daubechies e sono molto famose perché massimizzano il numero di momenti nulli. Nonostante questo possiamo notare dalla figura che non sono affatto lisce e regolari. Si può notare che le basi illustrate sono molto differenti dalle basi di Haar, o da quelle "a cappello messicano", poiché hanno caratteristiche diverse e sono utilizzate per applicazioni altrettanto differenti, ma permettono tutte l'analisi multirisoluzione desiderata attraverso la DWT, che in questo modo si dimostra uno strumento notevolmente malleabile a seconda delle esigenze.

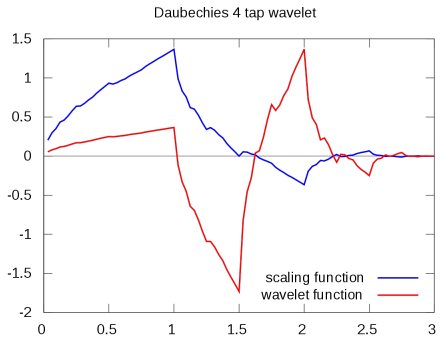


Figura 2.11: La scelta più famosa per le funzioni wavelet-madre (cioè la famiglia di wavelet) e wavelet-padre (cioè la famiglia di funzioni di scala) è quella delle wavelet di Daubechies.

Concludiamo questa sezione con un veloce confronto con l'espansione in serie di Fourier.

La principale differenza è quella di avere una diversa suddivisione dello spazio tempo-frequenza a causa della forma delle basi wavelet; per questo motivo abbiamo anche un'altra importante differenza, cioè la presenza necessaria della funzione di scala come ulteriore tipo di base. Per il resto la scomposizione tramite la trasformata wavelet ricalca da vicino la proiezione in sottospazi utilizzata nelle serie di Fourier. Questa particolare analisi multirisoluzione, però, porta grandi benefici e risulta utile in molteplici situazioni che approfondiremo nel terzo capitolo, e anche le implementazioni a livello computazionale risultano estremamente efficienti grazie all'utilizzo della Trasformata Wavelet Veloce che approfondiremo nel prossimo paragrafo.

in modo efficiente dal punto di vista computazionale tramite la Trasformata Wavelet Veloce.

2.5 Trasformata Wavelet Veloce

La *Trasformata Wavelet Veloce* (o FWT) è uno dei motivi principali dell'importanza delle wavelet nell'analisi delle immagini. Essa viene anche chiamata *Algoritmo a spina di pesce di Mallat* (per una più ampia trattazione vedere [9]). Riprendiamo in considerazione l'equazione 2.22 di "rifinimento" multirisoluzione ottenuta nei paragrafi precedenti:

$$\varphi(x) = \sum_n h_\varphi(n)\sqrt{2}\varphi(2x - n) \tag{2.35}$$

Scalando poi x di 2^j, traslandolo di k e ponendo m=2k+n, otteniamo:

$$\varphi(2^j x - k) = \sum_m h_\varphi(m \breve{} 2k)\sqrt{2}\varphi(2^{j+1} x \breve{} m) \tag{2.36}$$

È importante notare che il vettore di scala h_φ può essere considerato come il "peso" utilizzato per espandere $\varphi(2^j x \breve{} k)$ come somma delle funzioni di scala con j+1. Similmente per $\psi(2^j x \breve{} k)$ otteniamo:

$$\psi(2^j x - k) = \sum_m h_\psi(m \breve{} 2k)\sqrt{2}\varphi(2^{j+1} x \breve{} m) \tag{2.37}$$

Inizialmente consideriamo h_ψ e h_φ come coefficienti dell'espansione in serie di wavelet, e in un secondo momento ci ricondurremo invece ad un'analisi considerando f discreta, e in quel caso rappresenteranno invece i coefficienti DWT.

Consideriamo ora i coefficienti dell'espansione in serie di wavelet $d_j(k)$ che rappresentano le proiezioni della funzione sul sottospazio generato dalle wavelet; sostituendo nel prodotto scalare la definizione di wavelet otteniamo:

$$d_j(k) = \int f(x) 2^{j/2} \psi(2^j x - k) dx \tag{2.38}$$

Sostituendo ora l'equazione 2.28 di rifinimento multirisoluzione per le wavelet, ricavata precedentemente otteniamo:

$$d_j(k) = \int f(x) 2^{j/2} \left[\sum_m h_\psi(m-2k)\sqrt{2}\varphi(2^{j+1}x \breve{} m) \right] dx \tag{2.39}$$

Sapendo che il valore della sommatoria è finito, posso quindi scambiare integrale e sommatoria [4] e ottenere:

$$d_j(k) = \sum_m h_\psi(m-2k) \left[\int f(x) 2^{j+1/2}\sqrt{2}\varphi(2^{j+1}x \breve{} m) dx \right] \tag{2.40}$$

Vediamo quindi che l'intero integrale può essere riscritto come prodotto scalare della funzione reale f(x) per la funzione di scala, cioè rappresenta la proiezione di f sul sottospazio generato da quest'ultima. Notiamo quindi a questo punto che questa è proprio la definizione di $c_{j+1}(m)$ e lo andiamo a sostituire ottenendo un'importantissima relazione tra i coefficienti dei dettagli alla scala j e quelli di approssimazione alla scala j+1:

$$d_j(k) = \sum_m h_\psi(m-2k) c_{j+1}(m) \tag{2.41}$$

Riprendendo ora la definizione dei coefficienti c_j, rappresentanti le proiezioni di f(x) sul sottospazio generato dalla funzione di scala φ_j, possiamo ripetere le stesse operazioni scambiando nuovamente sommatoria e integrale ottenendo un risultato simile e altrettanto importante:

$$c_j(k) = \sum_m h_\varphi(m-2k) c_{j+1}(m) \tag{2.42}$$

Si possono trovare ovviamente le stesse relazioni per i coefficienti della DWT, i quali sono diversi dai precedenti ma possono essere sostituiti senza troppi problemi ai coefficienti c_j e d_j quando f(x) è discreta:

$$\begin{aligned} W_\psi(j,k) &= \sum_m h_\psi(m-2k) W_\varphi(j+1,m) \\ W_\varphi(j,k) &= \sum_m h_\varphi(m-2k) W_\varphi(j+1,m) \end{aligned} \tag{2.43}$$

Conoscendo quindi le relazioni tra i sottospazi V_j e W_j generati dalle funzioni di scala e dalle funzioni wavelet rispettivamente, e sapendo quindi che sono entrambi contenuti nel sottospazio V_{j+1}, queste due relazioni che descrivono le proiezioni proprio su quei sottospazi assumono un senso intuitivo: sia i dettagli che l'approssimazione della scala j possono essere ottenuti a partire dall'approssimazione alla risoluzione maggiore.

[4] Per ulteriori informazioni sul teorema che permette lo scambio vedere il teorema 1.38 in [4].

Possiamo infine riscrivere le ultime due relazioni sfruttando la definizione dell'operatore di convoluzione:

$$W_\psi(j,k) = h_\psi(-n) * W_\varphi(j+1,n)\Big|_{n=2k,k\geq 0} \qquad W_\varphi(j,k) = h_\varphi(-n) * W_\varphi(j+1,n)\Big|_{n=2k,k\geq 0} \tag{2.44}$$

Dove le convoluzioni vengono calcolate in n=2k, k$\geq$ 0.

Come verrà mostrato nell'esempio 2.1 valutare le convoluzioni per indici pari non negativi equivale a filtrare e sottocampionare di un fattore 2. Questo significa che, essendo il filtro h_ψ o h_φ composto da una decina di elementi, per ogni elemento della sequenza da analizzare (nel caso delle immagini potrebbero essere dei pixel) avrò una decina di somme e altrettante moltiplicazioni. Questo vuol dire che per una sequenza di lunghezza $M = 2^J$, il numero delle operazioni matematiche necessarie è dell'ordine di O(M). Perciò il numero di moltiplicazioni e addizioni è lineare rispetto alla lunghezza della sequenza di input. A questo punto possiamo facilmente paragonare la FWT alla Trasformata Veloce di Fourier, che richiede operazioni dell'ordine di $O(Mlog_2M)$, e vedere già un importante vantaggio nell'analisi Wavelet: essa richiederà una capacità computazionale minore. E questo su grandi moli di dati ha un'importanza notevole.

2.5.1 I vettori di scala e wavelet

Per comprendere meglio l'utilizzo della FWT, è importante capire cosa siano esattamente i filtri h_ψ o h_φ. Fondamentalmente sono il prodotto del ragionamento che avevamo fatto per ricavare le equazioni di rifinimento per la funzione di scala (2.22) e la funzione wavelet(2.28). Essi infatti sono i "pesi" che permettono di collegare i due tipi di funzione alla scala j, con la nuova funzione di scala alla risoluzione più alta, j+1.

Possiamo darne quindi una definizione sfruttando la delta di Dirac $\delta(n)$ e sostituendola all'input f(n)=$W_\varphi(j+1,n)$ per una scala j sufficientemente grande nell'equazione 2.43:

$$h(n) = \sum_n h_\psi(n-2k)\delta(n) = h_\psi(-n) * \delta(n)\Big|_{n=2k,k\geq 0} \tag{2.45}$$

Possiamo quindi definire i vettori di scala e wavelet come *risposta all'impulso* $\delta(n)$, poiché è proprio il risultato dell'analisi wavelet di un segnale composto da un impulso mostrato in figura 2.12 :

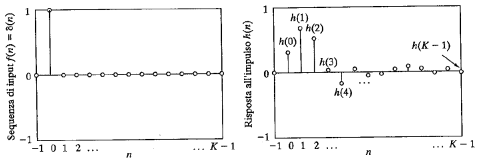


Figura 2.12: A sinistra è mostrata una sequenza discreta di impulsi unitari, mentre a destra la risposta del filtro all'impulso.

Da quest'analisi risultano, inoltre, essere più importanti anche delle funzioni di base wavelet in quanto rappresentano i filtri con i quali calcolare la convoluzione della funzione f(x). In effetti si può notare che questi filtri sono specifici delle diverse famiglie di wavelet perché specificano il rapporto tra le funzioni di base della famiglia al variare della risoluzione, perciò sono strettamente collegati alle basi utilizzate.

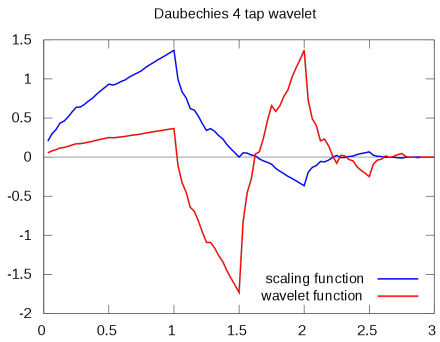
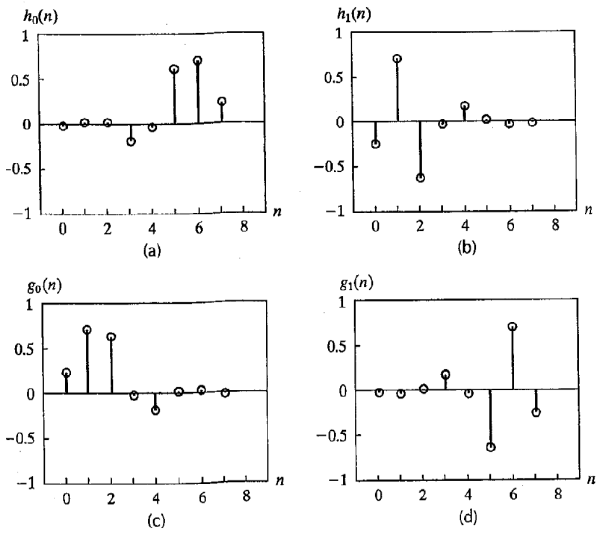


Figura 2.13: La scelta più famosa per le funzioni wavelet-madre (cioè la famiglia di wavelet) e wavelet-padre (cioè la famiglia di funzioni di scala) è quella delle wavelet di Daubechies.

La versione discreta della base di Daubechies, che abbiamo riportato per comodità nella figura 2.13, ha un corrispettivo filtro che, invece, è rappresentato nell'immagine 2.14. In essa i filtri h_j con j = 0,1 generano il banco dei filtri di analisi (vedremo come nei prossimi paragrafi), mentre g_j con j = 0,1 generano quello dei filtri di sintesi, ma come abbiamo precedentemente sottolineato, essendo ortonormali, i due banchi di filtri sono strettamente correlati come si può notare dalla figura. In generale abbiamo infatti che, se le basi costituiscono dei *filtri di ricostruzione perfetti*, cioè l'immagine può essere ricostruita in modo perfetto e se risultano basi ortonormali, risulta valida la relazione (per ulteriori dettagli vedere [7]):

$$g_i(n) = h_i(-n) \tag{2.46}$$



n	$g_0(n)$
0	0.23037781
1	0.71484657
2	0.63088076
3	−0.02798376
4	−0.18703481
5	0.03084138
6	0.03288301
7	−0.01059740

Figura 2.14: La relativa versione discreta della base precedente a 8 campionamenti. Maggiori sono i campionamenti della base considerati e maggiore è la precisione nell'analisi. A fianco troviamo i coefficienti del filtro ortonormale di Daubechies. (per ulteriori approfondimenti, vedere [5])

È importante notare quindi che tramite queste definizioni siamo riusciti a creare un *banco di filtri* h_ψ o h_φ, che può essere "iterato" per creare strutture a più stadi utili a calcolare i coefficienti DWT per due o più scale successive, cioè abbiamo la possibilità di generare l'intera piramide multirisoluzione.

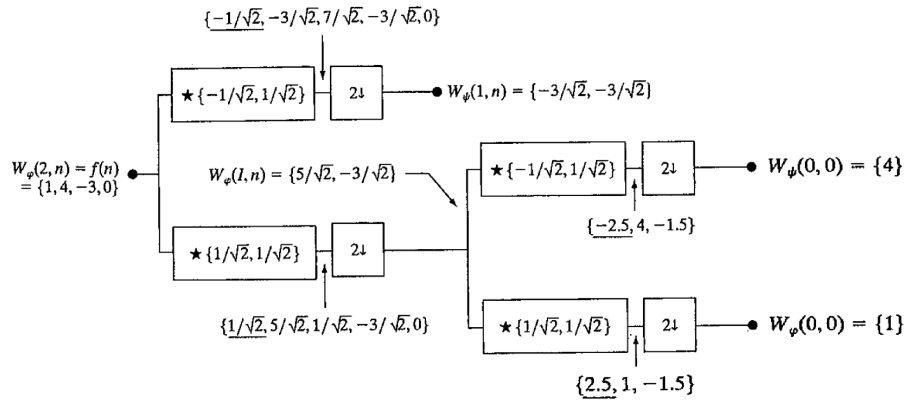
Esempio 2 *Per illustrare meglio questi concetti, si consideri la funzione discreta f(n)=1, 4, -3, 0. Calcoliamo ora la trasformata basandoci sulle funzioni wavelet e di scaling di Haar. Sfrutteremo ovviamente la Trasformata Wavelet Veloce e per questo utilizzeremo i filtri corrispondenti:*

$$h_\varphi(n) = \begin{cases} \frac{1}{\sqrt{2}} & n = 0, 1 \\ 0 & altrimenti \end{cases} \tag{2.47}$$

e

$$h_\psi(n) = \begin{cases} \frac{1}{\sqrt{2}} & n = 0 \\ -\frac{1}{\sqrt{2}} & n = 1 \\ 0 & altrimenti \end{cases} \tag{2.48}$$

Otteniamo quindi:



Dall'esempio mostrato si può notare che i coefficienti della scala più alti vengono assunti come campioni della funzione stessa, cioè $W_\varphi(J, N) = f(n)$, dove J è la scala più alta. Del resto sappiamo che la risoluzione di un segnale campionato è finita a causa del teorema di Nyquist. Infatti la finitezza nella frequenza di campionamento comporta inevitabilmente la perdita di alcune frequenze, quantificate dal seguente teorema:

Teorema 7 *Dato un segnale analogico s(t) la cui banda di frequenze sia limitata dalla frequenza f_M, e dato n∈ Z, il segnale s(t) può essere univocamente ricostruito a partire dai suoi campioni $s(n\triangle t)$ presi a frequenza $f_s = \frac{1}{\triangle t}$ se e solo se $f_S > 2f_M$.*

In questo modo la funzione f(n) può essere assunta come proiezione del segnale di partenza su un sottospazio V_J, a cui appartiene quindi f(n). A questo punto lo spettro della funzione originale è trasformato in due componenti mezza-banda: il sottospazio di wavelet W_{j-1} e il sottospazio di scala V_{j-1}. Il secondo banco di filtri a sua volta trasforma lo spettro e il sottospazio V_{j-1}, la mezza-banda più bassa nei sottospazi a quattro bande W_{j-2} e V_{j-2} con i rispettivi coefficienti DWT $W_{j-2}(n)$ e $V_{j-2}(n)$. Ovviamente il banco di filtri a due fasi si estende facilmente rispetto ad un qualsiasi numero di scale. In particolare solitamente, si scelgono 2^J campioni di f(x) e si impiegano P banchi di filtri per creare una FWT a P scale con scale J-1, J-2,. . . ,J-P.

La Trasformata Wavelet Veloce Inversa

Concludiamo la trattazione della FWT parlando della relativa trasformata inversa che ovviamente si basa anch'essa sui vettori di scala e wavelet h_ψ o h_φ. Come già notato, la ricostruzione perfetta richiede che:

$$g_i(n) = h_i(-n) \tag{2.49}$$

Per i=0,1. Cioè i filtri di analisi e di sintesi devono essere invertiti l'uno rispetto all'altro. La figura 2.15 illustra in dettaglio la struttura della trasformata inversa.

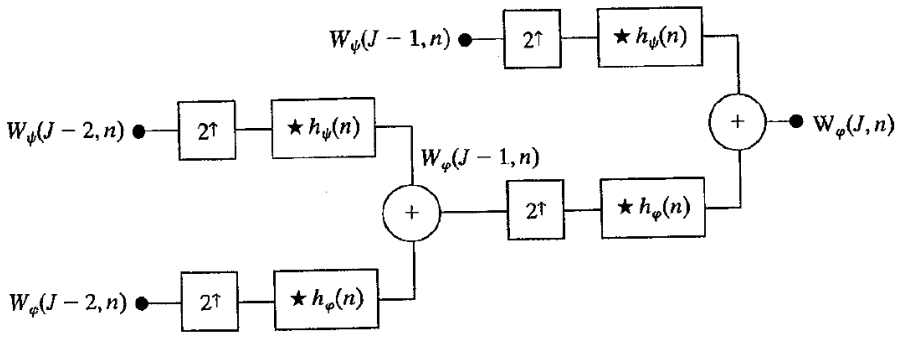


Figura 2.15: Banco di filtri di sintesi FWT^{-1} a due fasi o due scale.

2.6 Trasformate Wavelet in due dimensioni

Concludiamo finalmente il lungo capitolo sull'analisi tramite Wavelet con le trasformate wavelet di funzioni bidimensionali, che risultano essere semplicemente l'estensione della trasformata in una dimensione.

In particolare in due dimensioni è necessaria una funzione di scaling e tre wavelet bidimensionali ψ. Ognuna di queste è generata dal prodotto di due funzioni bidimensionali, escludendo prodotti che risultano monodimensionali (come $\varphi(x)\psi(x)$):

$$\begin{aligned} &\textit{Funzione di scaling separabile} : \varphi(x,y) = \varphi(x)\varphi(y) \\ &\textit{Wavelet separabili ``sensibili alla direzione''} : \\ &\psi^H(x,y) = \psi(x)\varphi(y) \\ &\psi^V(x,y) = \varphi(x)\psi(y) \\ &\psi^D(x,y) = \psi(x)\psi(y) \end{aligned} \tag{2.50}$$

Queste tre wavelet misurano variazioni nella funzione, cioè variazioni di intensità per le immagini, lungo direzioni diverse: $\psi^H(x,y)$ misura quelle lungo le colonne, cioè picchi orizzontali, $\psi^H(x,y)$ lungo le righe, $\psi^H(x,y)$ lungo la diagonale. Si definiscono perciò le funzioni di base traslate e scalate:

$$\varphi_{j,m,n}(x,y) = 2^{j/2}\varphi(2^j x - m, 2^j y - n)\psi^i_{j,m,n}(x,y) = 2^{j/2}\psi^i(2^j x - m, 2^j y - n), \quad i = \{H,V,D\} \tag{2.51}$$

Similmente al caso monodimensionale, la trasformata discreta wavelet di un'immagine

f(x,y) di dimensioni MxN risulta quindi

$$\begin{aligned} W_\varphi(j_0,m,n) &= \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\varphi_{j_0,m,n}(x,y) \\ W_\psi^i(j,m,n) &= \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\psi_{j,m,n}^i(x,y) \end{aligned} \tag{2.52}$$

Dove j_0 è una scala di partenza arbitraria e i coefficienti $W_\varphi(j_0,m,n)$ definiscono un'approssimazione di f(x,y) alla scala j_0, mentre quelli $W_\psi^i(j,m,n)$ aggiungono dettagli orizzontali, verticali e diagonali per le scale j$\geq j_0$.

La Trasformata Wavelet Discreta Inversa risulta:

$$\begin{aligned} f(x,y) =& \frac{1}{\sqrt{MN}} \sum_m \sum_n W_\varphi(j_0,m,n)\varphi_{j_0,m,n}(x,y) \\ &+ \frac{1}{\sqrt{MN}} \sum_{i=H,V,D} \sum_{j=j_0}^{\infty} \sum_m \sum_n W_\psi^i(j,m,n)\psi_{j,m,n}^i(x,y) \end{aligned} \tag{2.53}$$

Infine, come nella DWT 1-D, anche in due dimensioni possiamo utilizzare filtri digitali e sottocampionatori nella Trasformata Wavelet Veloce come mostrato nello schema 2.16. In particolare con funzioni wavelet e di scala bidimensionali separabili, prendiamo semplicemente la FWT 1-D delle righe f(x,y), seguita dalla FWT1-D delle colonne che ne risultano:

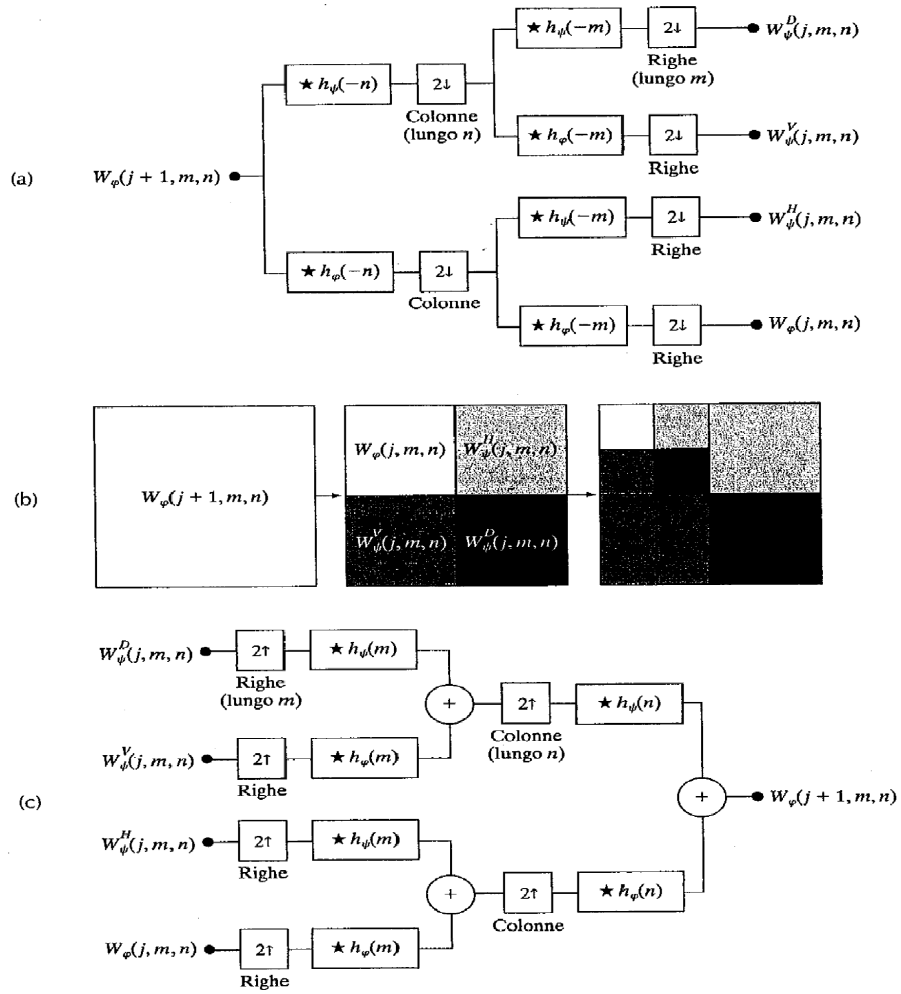


Figura 2.16: Banco di filtri a quattro bande bidimensionale per la codifica di un'immagine.

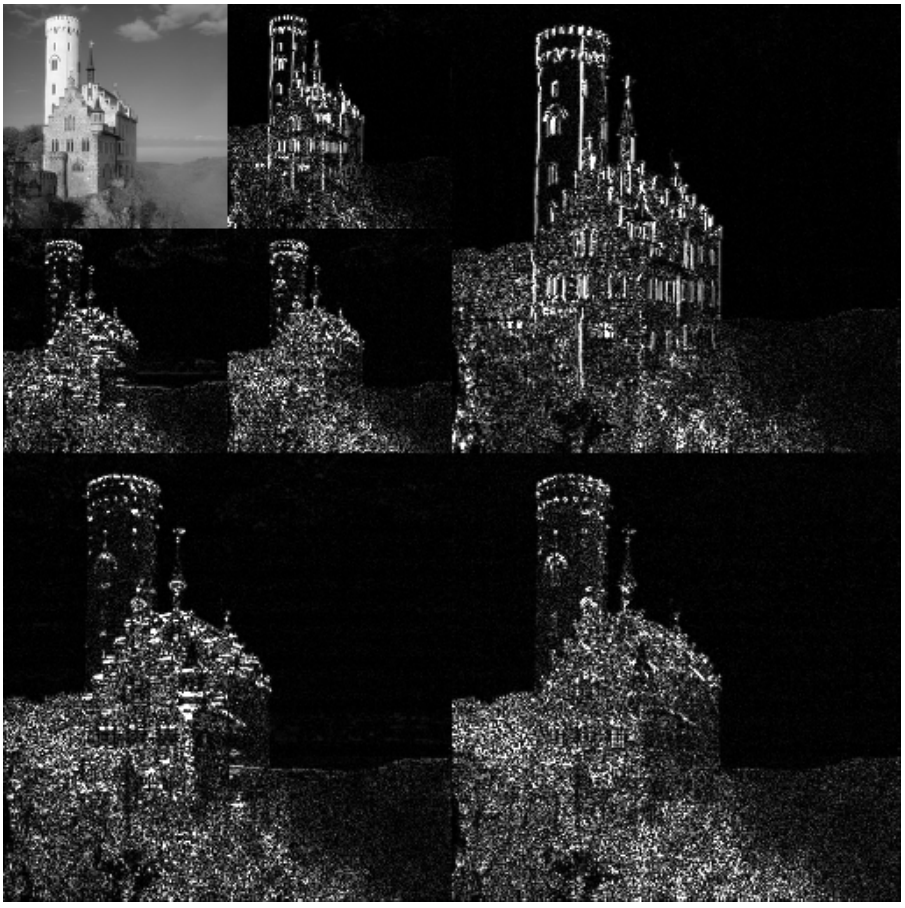


Figura 2.17: Un esempio di calcolo della FWT a due scale. Si possono notare distintamente le righe verticali nelle due immagini a sinistra, dovute proprio all'analisi delle variazioni lungo le righe (cioè picchi orizzontali come i contorni delle finestre).

Nel prossimo capitolo passeremo ad illustrare le numerose applicazioni e vantaggi di questo tipo di analisi, e andremo ad approfondire come venga utilizzata all'interno del formato JPEG2000.

Capitolo 3

Applicazioni e Vantaggi dell'Analisi Wavelet

In questo capitolo riprenderemo il discorso anticipato nel primo capitolo riguardo la compressione delle immagini e introdurremo la codifica JPEG2000 senza però approfondire gli aspetti tecnici e gli standard utilizzati nelle diverse fasi del processo, poiché non è l'obiettivo di questa tesi. Confronteremo invece la compressione di immagini nel formato JPEG, basato sulla trasformata discreta del coseno (DCT), con quella nel formato JPEG2000 per concludere identificando possibili vantaggi e campi di applicazione di quest'ultimo.

3.1 Il formato JPEG2000

Come abbiamo già spiegato all'interno del primo capitolo, per un'efficiente compressione JPEG è necessario suddividere l'immagine in diverse parti[1]. Questo processo data la natura delle trasformate wavelet non è necessario nel formato JPEG2000. In questo caso, infatti, le trasformate wavelet sono già fortemente localizzate e sfruttando anche l'analisi multirisoluzione e la trasformata wavelet veloce (FWT) riescono ad essere molto efficienti computazionalmente.

In entrambi i formati di compressione presi in esame le trasformate si rivelano molto utili perché raccolgono in un numero limitato di coefficienti la maggior parte delle informazioni visive e i coefficienti rimanenti possono essere ridotti o anche eliminati senza introdurre distorsioni eccessive all'interno dell'immagine.

[1]La suddivisione in sottoimmagini è molto utile soprattutto dal punto di vista computazionale e del trasferimento. Infatti suddividendo l'immagine in diverse parti, ognuna di esse può essere inviata in pacchetti completi ed essere poi elaborata tramite architetture parallele come accade all'interno dei processori più recenti *multicore*.All'interno delle schede grafiche, spesso utilizzate per questo tipo di elaborazioni, i processori grafici GPU spesso contenengono migliaia di processori che lavorano in parallelo.

Le basi wavelet

Come abbiamo già spiegato le basi nella Trasformata Wavelet (DWT) non sono fissate e per questo ne esistono diverse topologie, da quelle elementari e più "vecchie" storicamente fino a quelle più moderne elaborate e complesse. Questa grande varietà è dovuta al fatto che la scelta delle basi influenza ampiamente tutti gli aspetti della codifica: dalla complessità computazionale delle trasformate alla capacità del sistema di comprimere e ricostruire le immagini, fino alla possibilità di estrapolazione delle strutture e contorni all'interno dell'immagine. In particolare la capacità di conservare l'informazione in un ristretto numero di coefficienti della trasformata determina le prestazioni in termini di compressione e qualità di ricostruzione.

Fondamentalmente ci sono due tipi di basi wavelet: quelle ortogonali e quelle biortogonali. Il secondo tipo, che non è stato ancora trattato, è una generalizzazione del primo, che permette una maggior libertà nella costruzione della base perché i filtri di analisi e sintesi del segnale sono differenti[2], ed è proprio quello che viene sfruttato nel formato JPEG2000.

In particolare gli standard relativi a quest'ultimo impongono che vengano utilizzati due tipi di wavelet base: uno per la compressione *lossless* e uno per quella *lossy*, di cui abbiamo già parlato. Infatti possono essere sfruttate sia le trasformate reversibili "da intero a intero", sia quelle non reversibili "da reale a reale". In realtà come si vede dalla figura 3.2 sarebbe più corretto parlare di numeri *pseudoreali* in quanto, anche se presi con una precisione sufficientemente grande, i numeri reali non sono mai rappresentabili in modo esatto all'interno di un calcolatore con memoria finita.

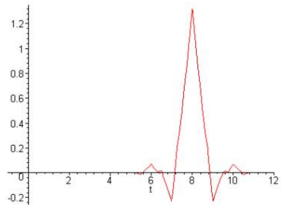


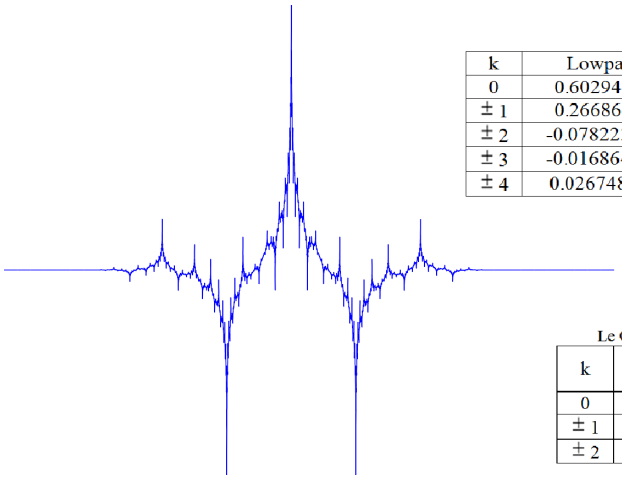
Figura 3.1: Wavelet corrispondente al filtro CDF 9/7 utilizzato nello standard JPEG2000 per la compressione lossy.

Poiché la compressione senza perdita di informazioni (lossless) richiede che nessun dato venga perduto a causa di arrotondamenti, è necessaria una trasformata wavelet che usa solo filtri razionali. Viceversa, la compressione con perdita (lossy) permette che qualche dato venga perduto, è per questo è concesso l'utilizzo di trasformate wavelet non reversibili, cioè filtri con coefficienti non razionali.

Indipendentemente dal tipo di compressione le basi utilizzate sono chiamate *Funzioni Wavelet di Cohen-Daubechies-Feauveau*(o CDF) ma la compressione lossless si differenzia

[2] Le funzioni di espansione biortogonali sono wavelet alle quali è associata una trasformata invertibile, e per questo è possibile una ricostruzione perfetta, ma hanno la peculiarità di non essere necessariamente ortogonali. Questo significa che nel caso biortogonale esistono due differenti funzioni wavelet e due di scala poiché quelle di analisi differiscono da quelle di ricostruzione del segnale, ma sono legate da una relazione più complessa e meno restrittiva rispetto a quella delle basi ortogonali: $g_0(n) = (-1)^{n+1}h_1(n)$, e $g_1(n) = (-1)^n h_0(n)$ e per questo si dicono *modulati a croce*. Per ulteriori informazioni sulle basi wavelet biortogonali, vedi [10]

dall'altra per la lunghezza del filtro che risulta *5/3* nello standard JPEG2000, dove il primo numero si riferisce a quella del filtro di analisi passabasso (o relativo alla funzione di scala) e a quella del filtro di sintesi passaalto (o relativo alla funzione wavelet), mentre il secondo si riferisce a quella del filtro di analisi passaalto e a quella del filtro di sintesi passabasso (sappiamo infatti che i filtri biortogonali sono *modulati a croce*, a causa della loro proporzionalità "incrociata"). Nel caso della compressione lossy, la lunghezza dei filtri mostrati in figura 3.1 risulta invece 9/7 e i coefficienti sono reali (vedi figura 3.2).



Daubechies 9/7 Analysis Filter Coefficients

k	Lowpass Filter (h_o)	Highpass Filter (g_o)
0	0.6029490182363579	1.115087052456994
± 1	0.2668641184428723	-0.5912717631142470
± 2	-0.07822326652898785	-0.05754352622849957
± 3	-0.01686411844287495	0.09127176311424948
± 4	0.02674875741080976	

Le Gall 5/3 Analysis Filter Coefficients

k	Lowpass Filter (h_o)	Highpass Filter (g_o)
0	6/8	1
± 1	2/8	½
± 2	-1/8	

Figura 3.2: A sinistra: La funzione wavelet 5/3 di Cohen-Daubechies-Feauveau con 2/2 momenti nulli utilizzata nello standard JPEG2000. A destra: I valori utilizzati come filtri wavelet e di scala per le funzioni di base nella compressione lossless (sopra) e lossy (sotto). È importante notare che i coefficienti della wavelet 5/3 sono tutti numeri razionali, la cui rappresentazione binaria esatta è realizzabile (a differenza dei coefficienti nella tabella sopra dei quali si può considerare solo un numero limitato di cifre più o meno grande a seconda della precisione richiesta).

Come si può vedere nella figura 3.3 la scelta della base può influenzare anche del 10 % la capacità di compressione dell'algoritmo. A seconda dei filtri scelti il numero di operazioni può variare notevolmente (nel nostro caso si passa da 4 operazioni nel caso delle wavelet di Haar, fino a 28 nel caso di quelle biortogonali CDF 17/11) per ogni coefficiente e per ogni livello di scomposizione (scala). Al crescere della complessità computazionale (e quindi alla lunghezza del filtro) cresce anche la capacità di "compattare" l'informazione e poter quindi azzerare un numero maggiore di coefficienti senza subire maggiori distorsioni.

Prima di passare alla delicata fase della quantizzazione, è utile notare la soluzione utilizzata nel particolare caso che si verifica vicino ai limiti del segnale, come avviene sui bordi dell'immagine. In questa situazione lo standard JPEG2000, similmente a quello JPEG, sfrutta un'estensione simmetrica, la quale aggiunge un'immagine speculare del se-

Wavelet	Tap del filtro (scaling + wavelet)	Coefficienti azzerati
Haar	2 + 2	33.8%
Daubechies	8 + 8	40.9%
Symet	8 + 8	41.2%
Biortogonali	17 + 11	42.1%

Figura 3.3: In questa tabella è mostrata la trasformata wavelet, il numero di tap del filtro, cioè la sua lunghezza, e il numero di coefficienti azzerati impostando un valore di soglia 1.5. Le trasformate symlet sono simili a quelle di Daubechies, ma con un grado maggiore di simmetria, mentre le trasformate biortogonali coincidono con quelle CDF trattate.

gnale all'esterno dei bordi in modo tale da evitare l'introduzione di errori particolarmente rilevanti nel calcolo dei coefficienti.

3.1.1 Quantizzazione

Dopo che ogni livello di scomposizione è stato calcolato e analizzato tramite la Trasformata Wavelet Veloce, il numero totale dei coefficienti della trasformata è uguale al numero di campioni nell'immagine originale, ma l'informazione visiva si concentra solo in un numero ristretto di coefficienti. Nel caso della compressione lossless avremo quindi una matrice di coefficienti razionali e solo una minor parte sarà esattamente zero e per questo potremo evitare di codificarli. Nel caso della compressione lossy, invece, la matrice sarà composta da numeri pseudoreali, tutti diversi da zero che però tramite l'operazione di quantizzazione posso ricondurre a numeri arrotondati, i quali possono avere una rappresentazione binaria molto più compatta a seconda delle richieste di compressione. Il processo di quantizzazione infatti genera una perdita di qualità e di informazioni all'interno dell'immagine e per questo motivo è un'operazione non reversibile.

Come nel caso dei numerosi filtri wavelet possibili, per ottenere il miglior compromesso tra la perdita di qualità e il rapporto di compressione, anche in questo caso sono stati sviluppati diversi quantizzatori.

Come prima operazione viene scelto uno specifico intervallo, chiamato *dead zone*, all'interno del quale, ogni coefficiente viene posto a zero; in un secondo momento i coefficienti pseudoreali vengono arrotondati. Spesso, però, queste due operazioni vengono eseguite contemporaneamente tramite una curva di quantizzazione come quella in figura 3.4. Questo è il caso che avevamo già visto all'interno dell'esempio 1.

Nello standard JPEG2000, però, la quantizzazione risulta un po' più ottimizzata e il suo "passo" non è sempre costante. Esso dipende infatti dal numero di bit assegnati all'esponente (ϵ_b) e da quelli assegnati alla mantissa (μ_b) dei coefficienti del sottointervallo

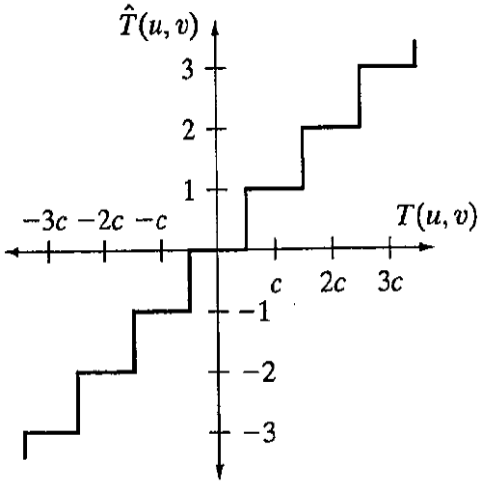


Figura 3.4: Un esempio di curva di quantizzazione della codifica *threshold*(cioè a soglia. Da notare la *dead zone* ampia esattamente quanto gli altri intervalli.

b da rappresentare. In particolare un coefficiente $a_b(u,v)$ della sottobanda b è quantizzato al valore $q_b(u,v)$ utilizzando:

$$q_b(u,v) = sign\left[a_b(u,v)\right] \cdot floor\left[\frac{|a_b(u,v)|}{\triangle_b}\right] \tag{3.1}$$

Dove R_b risulta il *range nominale dinamico*[3]. In questo modo a seconda delle capacità e dell'architettura del calcolatore avrò una quantizzazione sempre ottimale perché adattabile.

3.1.2 Riordinamento dei dati

Anche il riordinamento dei coefficienti trasformati rimane perfettamente in linea con l'idea dell'analisi multirisoluzione e della suddivisione dell'informazione basata sulla sua importanza. Infatti dopo aver quantizzato i coefficienti, questi ultimi vengono sistemati in blocchi rettangolari chiamati *blocchi di codice*, che vengono analizzati singolarmente, sempre per favorire la computazione parallela che avviene all'interno dei moderni processori *multicore*.

Per sfruttare poi i vantaggi dell'analisi multirisoluzione i diversi blocchi vengono elaborati un piano alla volta a partire da quello relativo al bit più significativo (che ovviamente contenga almeno un elemento diverso da zero) (vedi figura 3.5).

[3]Il *range nominale dinamico* della sottobanda b è la somma del numero dei bit utilizzati per rappresentare l'immagine originale e il numero di bit dell'*analysis gain*(che letteralmente significa "guadagno di analisi") per la sottobanda b. (Per ulteriori informazioni vedere [2])

Come mostrato nella figura 3.6, anche all'interno dello stesso piano, si distinguono tre gruppi di bit in base alla loro importanza; ogni bit è contenuto in uno solo di essi ed ogni gruppo viene codificato in un ordine preciso: prima quello per il *raffinamento della magnitudo*, poi quello per la *propagazione del significato*, e infine quello per la *normalizzazione*. Per capire meglio questi tre passaggi definiamo prima di tutto un coefficiente come *significativo* se esiste almeno un bit non nullo tra i suoi bit corrispondenti codificati nei piani di bit precedenti, altrimenti il coefficiente è detto *non significativo*. Questo è dovuto al fatto che, se un coefficiente è significativo (cioè grande numericamente), avrà automaticamente un 1 tra i bit più significativi (MSB). Una volta che il coefficiente è definito significativo, tutti i suoi coefficienti verranno considerati tali e per questo codificati con la massima priorità. Possiamo ora cercare di capire meglio i tre gruppi di bit (per una spiegazione più completa vedere [11]):

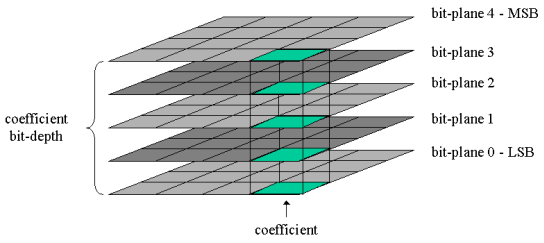


Figura 3.5: Esempio di codifica in piani di bit dei coefficient wavelet con una profondità di bit pari a 5.

1. Il gruppo per il *raffinamento della magnitudine* è utilizzato per migliorare la grandezza del coefficiente corrispondente al bit che sta venendo codificato, e quindi migliora la precisione di quel coefficiente specificandone un ulteriore bit. Per questo motivo il bit appartiene a questo gruppo solo se appartiene ad un coefficiente già identificato come significativo.

2. Il gruppo per la *propagazione del significato* raccoglie informazioni per propagare la significatività dei bit. In particolare se un bit diventa significativo, la probabilità che i coefficienti adiacenti ad esso diventino significativi diventa di volta in volta maggiore al procedere della codifica. Per questo quei bit sono codificati prima di quelli del gruppo di normalizzazione: il fatto che siano nulli è di per sé molto significativo perché significa che c'è un certo distacco tra i due bit adiacenti e quindi la loro codifica è necessaria per mantenere i dettagli dell'immagine. Appartengono a questo gruppo tutti quei bit non significativi, che sono adiacenti ad almeno un bit significativo.

3. Nel gruppo di *normalizzazione* tutti i restanti bit (non significativi e lontani da quelli significativi) vengono codificati. Questo gruppo è codificato per ultimo perché ad esso appartengono in linea di massima coefficienti di zone omogenee.

Dopo aver codificato nel rispettivo ordine i tre gruppi, è necessario raccogliere i diversi blocchi di codice rettangolari in cui avevamo suddiviso l'immagine inizialmente,

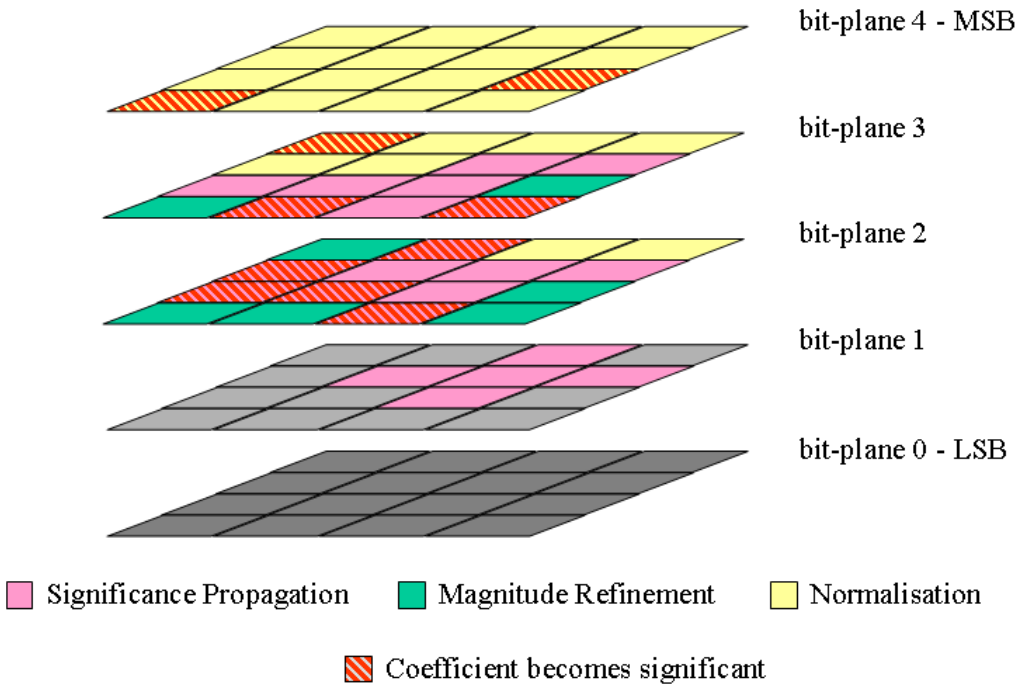


Figura 3.6: Esempio di codifica dei bit all'interno dei diversi piani in base alla loro importanza.

mantenendo rigorosamente i piani di bit e i gruppi. In questo modo genero dei "macro-piani", denominati *layer*, di cui sfrutto la suddivisione in "macrogruppi" per codificarli. In particolare i diversi layer vengono infine suddivisi in *pacchetti* che rappresentano l'unità fondamentale del mio flusso di dati. L'ultima operazione della codifica nel formato JPEG2000 è la compressione lossless del flusso binario sfruttando lo cosiddetta *codifica aritmetica*.

3.1.3 Codifica Aritmetica

Nel formato JPEG2000, sia nella compressione lossless che quella lossy, la codifica del codice binario viene eseguita tramite un algoritmo leggermente diverso da quello descritto nel Capitolo 1. Questo algoritmo viene chiamato "Codifica Aritmetica". L'idea è simile alla codifica di Huffman già descritta perché si cerca di codificare con un numero di bit minore la sequenza in cui compaiono i simboli con probabilità più alta. Questo diverso tipo di codifica, le cui basi sono state sviluppate da Elias intorno al 1960, è utilizzato solo nei formati più recenti come gli MP4 e, appunto, Jpeg2000. Si basa sull'utilizzo dei numeri decimali e per questo non esiste una corrispondenza univoca tra i simboli della sorgente e la loro codifica ("codeword") perché a un'intera sequenza di simboli è assegnabile un qualunque valore compreso in un intervallo reale (anche se solitamente si sceglie

quello con la rappresentazione binaria più breve). La "codeword" è sempre compresa tra 0 e 1 e al crescere del numero dei simboli del messaggio, l'intervallo utilizzato per rappresentare quest'ultimo (all'interno del quale scegliere la codeword) diventa sempre più piccolo, e quindi necessita di un numero crescente di bit. Se ad esempio bisogna codificare una sequenza di cinque simboli $a_1a_2a_3a_3a_4$, a partire da una sorgente nella quale sappiamo essere presenti quattro soli simboli (uno infatti è ripetuto), dapprima si assume che il messaggio occupi l'interno intervallo [0,1), poi questo intervallo viene suddiviso in due regioni di dimensioni proporzionali alla probabilità di ogni simbolo. Una volta, quindi, che si è individuato l'intervallo corrispondente al primo simbolo nella sequenza a_1, si riespande il nuovo intervallo e lo si suddivide mantenendo le stesse proporzioni di prima e si ripete l'operazione. In questo modo otterremo alla fine un intervallo che è tanto più largo quante più volte è presente il simbolo più probabile.

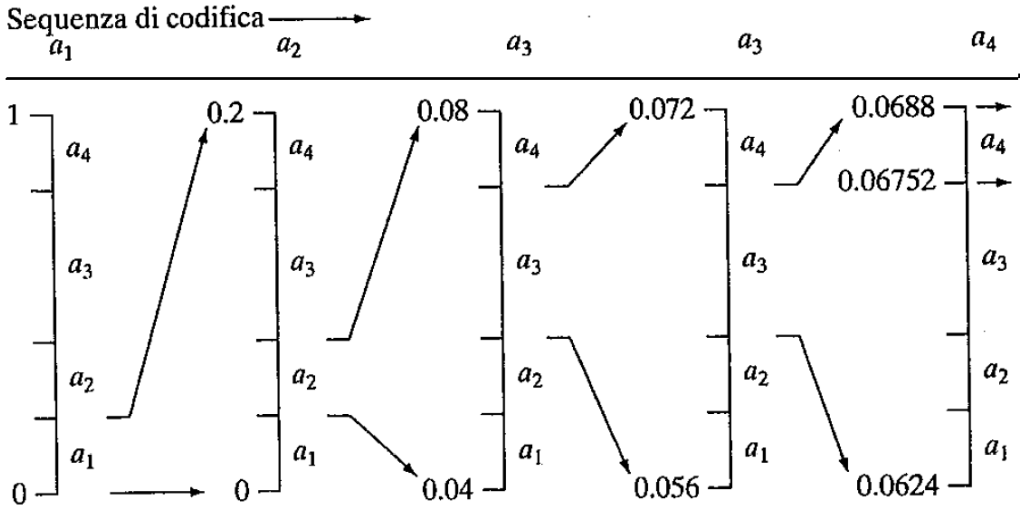


Figura 3.7: Esempio di codifica aritmetica di un codice con un dizionario composto da quattro *codeword* (cioè parole di codice), una delle quali viene ripetuta.

Dopo aver chiarito quindi tutti i passaggi della codifica nel formato JPEG2000, potremo facilmente dedurre che l'algoritmo necessario alla decodifica consiste nel ripercorrere le diverse fasi in ordine inverso. Ora invece chiariremo i vantaggi di questo tipo di codifica rispetto a quello basato sulle trasformate di Fourier che fornisce le basi per il formato predecessore, cioè il JPEG.

3.2 Vantaggi della Codifica JPEG2000

Cerchiamo ora di capire meglio i veri motivi che ci hanno portato a sviluppare questo tipo di algoritmo JPEG2000 e le trasformate Wavelet. I vantaggi infatti sono numerosi e cercheremo di riassumerli brevemente.

3.2.1 Analisi Multirisoluzione

L'obiettivo principale verso il quale abbiamo indirizzato tutta la nostra analisi è sicuramente l'analisi multirisoluzione, cioè la possibilità di avere un'immagine codificata mettendo in ordine i dati basandosi non sull'ordine spaziale (come avviene in molti altri formati compreso il JPEG) ma sulle informazioni contenute: partendo da quella più importante che contiene quelle fondamentali e imprescindibili che danno i contorni di un'immagine, fino ad arrivare all'informazione più ampia che permette la distinzione dei dettagli più fini.

Del resto, anche storicamente, quest'idea era già nata prima della nascita delle trasformate wavelet ed era utilizzata indipendentemente in numerosi campi: nella matematica pura (la risoluzione di Calderòn dell'identità nell'analisi armonica), nella fisica (stati coerenti per il gruppo-(ax+b) nella meccanica quantistica di Aslaken e Klauder (1968), collegato ad esempio all'atomo di idrogeno Hamiltoniano da Paul (1985)),nell'ingegneria (i filtri QMF di Esteban e Galland (1977), nell'analisi dei dati sismici (con J.Morlet (1983)), fino ad arrivare all'ingegneria elettrica (la proprietà di ricostruzione esatta che abbiamo visto nel capitolo 2 di Smith,Barnwell e Vetterli (1986)).

In seguito una sintesi dei diversi approcci si è rivelata molto fertile per tutti i campi correlati. In particolare l'analisi wavelet può essere utilizzata nell'analisi dei segnali musicali, onde elettromagnetiche come quelle delle immagini, e in generale è utile per studiare qualunque tipo di onde. Per permettere il suddetto ordinamento dell'informazione abbiamo fatto ricorso alla trasformata Wavelet, la quale, senza ulteriori elaborazioni, suddivide e analizza un'immagine in diverse frequenze, cioè a diverse risoluzioni, come mostrato in figura 3.8. In questo modo ritroviamo diversi piani, ognuno contenente dettagli diversi da ogni altro e, a seconda del piano in analisi, potremo riuscire ad analizzare gli schemi macroscopici o microscopici all'interno della figura. E questo ad esempio è molto utile nella "computer vision", la quale spesso basa i suoi studi sul riconoscimento dei contorni.

Un'altra applicazione molto utile dell'analisi multirisoluzione è la possibilità di trasmettere solo una parte dell'immagine senza ulteriori elaborazioni solitamente necessarie. Infatti la trasmissione è notevolmente malleabile a seconda della capacità di trasmissione e di calcolo del ricevente poiché essa può essere liberamente tagliata in un qualsiasi punto a seconda del grado di compressione desiderato. Infatti, essendo le informazioni più importanti ordinate sempre all'inizio del flusso di dati, il ricevente avrà sempre la possibilità di osservare ogni regione dell'immagine alla risoluzione desiderata. Come mostrato nella figura 3.9, il formato JPEG2000 permette comunque la codifica di intere regioni specificate all'inizio del flusso di dati: in questo modo posso avere la certezza che quelle regioni verranno codificate a piena risoluzione prima del resto dell'immagine.

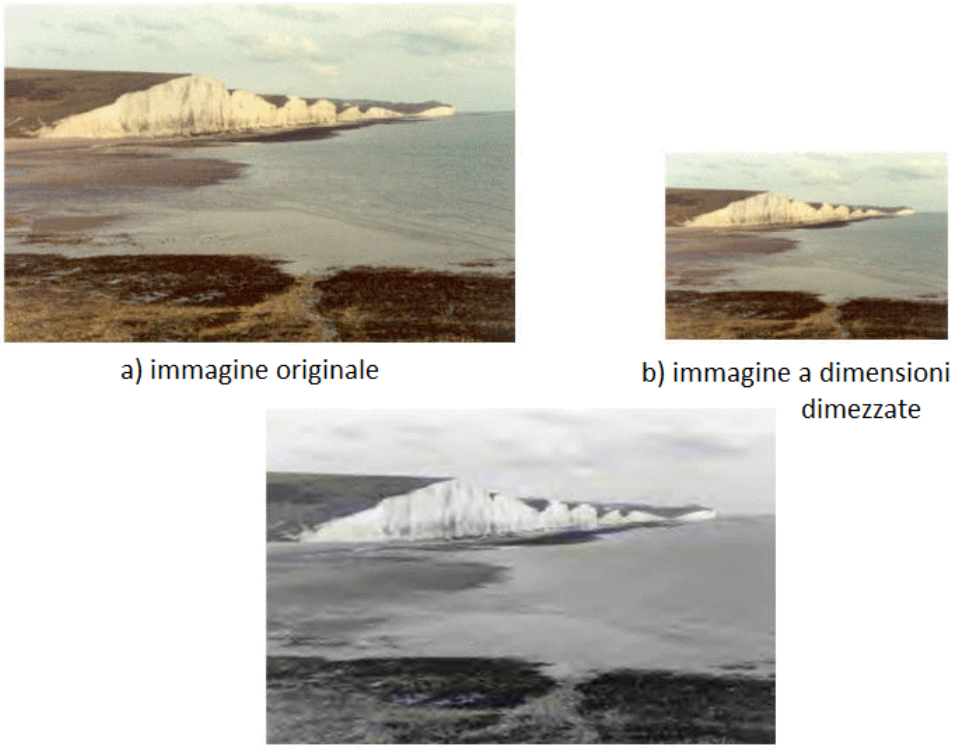


Figura 3.8: Esempio di codifica aritmetica di un codice con un dizionario composto da quattro *codeword* (cioè parole di codice), una delle quali viene ripetuta.

Questa capacità ha potenzialità enormi nel campo della fisica, dell'astrofisica, della medicina, dell'elaborazione grafica, video e molti altri. Infatti in questi campi si ha spesso a che fare con immagini ad altissima risoluzione, come quelle prodotte all'interno degli acceleratori di particelle, e per l'elaborazione sono spesso richiesti calcolatori estremamente potenti: per questo motivo anche la semplice visualizzazione di un'immagine richiede capacità computazionali notevoli. Come mostrato nella figura 3.10, l'analisi multirisoluzione risolve questo problema grazie alla possibilità di accedere a specifici dettagli dell'immagine senza ulteriori elaborazioni; dà inoltre la possibilità di accedere ad una copia dell'immagine con una risoluzione minore più facile da gestire ed elaborare, e in seguito applicare tutta l'elaborazione all'immagine originale sfruttando la potenza computazionale del

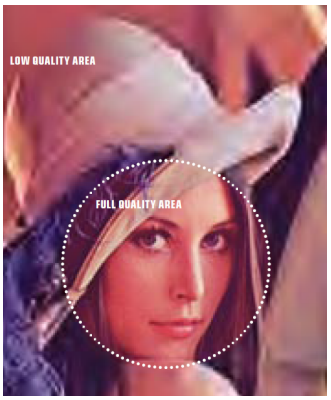


Figura 3.9: Nell'immagine si distingue visibilmente la migliore qualità nella resa del volto.

server che gliel'ha inviata.

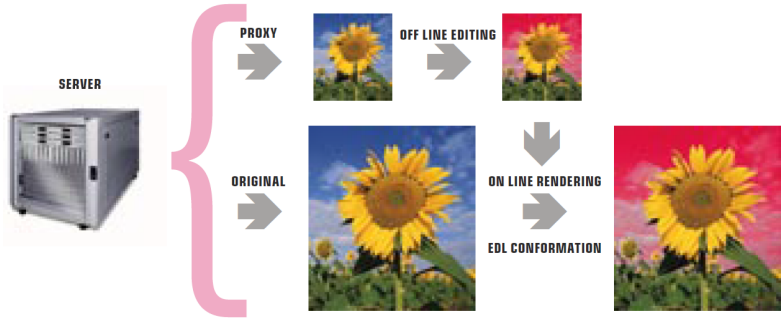


Figura 3.10: Nell'immagine sono mostrate le operazioni legate all'elaborazione di un'immagine memorizzata su un computer diverso da quello utilizzato per l'analisi.

Questa capacità è possibile anche grazie alla suddivisione e riorganizzazione in gruppi che abbiamo discusso nel paragrafo 3.1.2. Al suo interno infatti avevamo parlato dei piani di bit, divenuti poi layer, all'interno dei quali sono contenuti i bit più significativi. Perciò, quando vado a prelevare un'immagine e non ho una banda di trasmissione sufficientemente potente da poter caricare l'immagine originale, posso comunque conservare buona parte dei dettagli evitando eccessive distorsioni semplicemente troncando il flusso di dati e ricostruendo in questo modo solo la parte più significativa del singolo coefficiente. Questa possibilità si rivela molto utile anche nel campo della trasmissione di flussi di dati che giungono ad utenti diversi come può essere in ambiente WEB, nella televisione, o in un comune streaming di video, musica ed immagini. L'analisi multirisoluzione permette infatti che, una volta trasmesso un flusso di dati comune, il singolo ricevente possa scegliere, in base alla capacità di ricezione e alla potenza computazionale, la quantità e la qualità dei dati che vuole decodificare (figura 3.11).

Come abbiamo già spiegato nel paragrafo 2.5, dal punto di vista computazionale è importante infine notare che l'analisi multirisoluzione, attraverso la Trasformata Wavelet Veloce, porta ad un tempo necessario per il calcolo che cresce più lentamente della Trasformata Discreta del Coseno Veloce: il secondo è infatti maggiore del primo di un fattore pari al $log_2 N$, dove N è il numero di coefficienti da codificare, cioè i pixel. Anche questo dato ha grande rilevanza, soprattutto su grandi moli di dati.

3.2.2 Trasformate Wavelet

Dopo l'analisi multirisoluzione un altro aspetto molto importante della codifica JPEG2000 è costituito dall'analisi tramite Trasformate Wavelet. Essa infatti costituisce un particolare tipo di approccio allo studio dell'immagine senza mai specificare alcuna base specifica. Questa peculiarità rende le trasformate wavelet estremamente malleabili e adattabili

Figura 3.11: Esempio del miglioramento progressivo della qualità all'aumentare delle informazioni codificate.

al tipo di analisi che viene richiesto. In particolare le basi wavelet possono essere infinite e devono rispettare solo un numero molto limitato di restrizioni. Per questo motivo abbiamo le trasformate più veloci dal punto di vista computazionale, quelle più efficienti dal punto di vista del rapporto di compressione, quelle più efficaci in base all'analisi richiesta, come nel caso delle trasformate di Gabor molto potenti nel riconoscimento dei bordi (vedi figura 3.12).

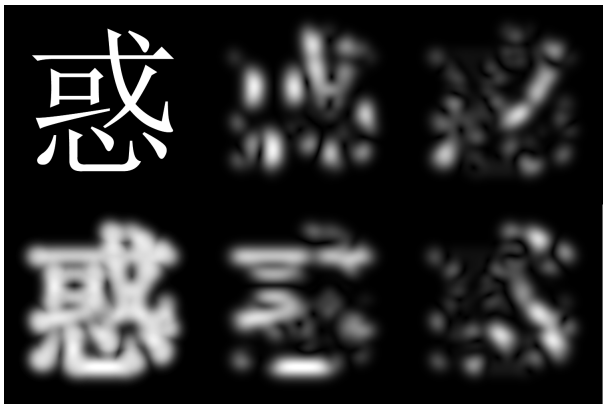


Figura 3.12: Il filtro di Gabor può analizzare le variazioni rispetto a diversi angoli. In questo caso sono mostrate sulla destra quelle a 0°, 45°, 90°, 135°. Le immagini sulla sinistra costituiscono invece l'immagine originale (sopra) e la sovrapposizione delle quattro immagini analizzate (sotto).

Un altro notevole vantaggio di queste basi è l'essere estremamente localizzate. In questo modo infatti, come abbiamo già mostrato abbiamo una suddivisione del piano tempo-frequenza che può essere mirata a determinate frequenze, e non è costretta perciò

ad essere omogenea come nel caso della trasformata del Coseno. Questo inoltre genera un altro enorme vantaggio in fase di compressione.

Le trasformate Wavelet infatti, grazie alla loro natura strettamente localizzata, eliminano i cosiddetti *Artefatti a Blocchi*. Queste particolari distorsioni dell'immagine dovute alla compressione sono generate dalla codifica JPEG a causa della suddivisione dell'immagine in blocchi 8x8. In particolare questa suddivisione crea una forte discontinuità sui bordi a causa del fatto che la Trasformata Discreta del Coseno viene calcolata su ogni singolo quadrato separatamente e questo rende l'immagine *omogenea a blocchi*.

Figura 3.13: A sinistra: l'immagine originale. Al centro: la stessa immagine compressa nel formato JPEG. A destra: l'immagine compressa nel formato JPEG2000. Gli *artefatti di compressione* ben visibili nell'immagine centrale sul cielo e sull'acqua sono molto meno evidenti con la compressione JPEG 2000.

Esistono poi anche altri tipi di distorsioni, chiamati *artefatti di compressione*, che vengono generati al momento della quantizzazione. Infatti l'eliminazione di una grossa parte dei coefficienti può portare a notevoli problemi all'immagine spesso molto visibili, come nel caso dell'immagine 3.13. La loro forma è spesso determinata dalle basi utilizzate all'interno delle due diverse trasformate. Infatti, mentre nel caso della DCT le basi sono funzioni sinusoidali di diversa frequenza e ampiezza che si propagano almeno per tutto il blocco 8x8, nel caso delle trasformate Wavelet la base può essere scelta tra diverse opzioni ma rimane comunque strettamente localizzata: l'intervallo di pixel in cui si propaga risulta infatti deciso dalla scala alla quale si opera. Nell'esempio 3 mostreremo meglio questo concetto.

Esempio 3 *Consideriamo la base di Haar ed analizziamo il segnale di un impulso, cioè (1,0,0,0). Poiché la base di Haar risulta (1,1,1,1), (1,1,-1,-1), (1,-1,0,0), mentre quella della trasformata discreta del coseno risulta (1,1,1,1), (1,0,-1,0) e (1,-1,1,-1), allora*

avremo le seguenti scomposizioni:

$$(1,0,0,0) = \frac{1}{4}(1,1,1,1) + \frac{1}{4}(1,1,-1,-1) + \frac{1}{2}(1,-1,0,0) \qquad \textit{Haar DWT} \tag{3.2}$$

$$(1,0,0,0) = \frac{1}{4}(1,1,1,1) + \frac{1}{2}(1,0,-1,0) + \frac{1}{4}(1,-1,1,-1) \qquad \textit{DFT} \tag{3.3}$$

La DWT dimostra in questo modo di essere localizzata poichè il primo termine (1,1,1,1) fornisce il valore medio, il secondo pone il segnale nella parte sinistra del dominio, e il terzo lo sposta nella parte più a sinistra (1,1,–1,–1) dello spazio già selezionato. Perciò, troncando ad uno qualunque dei tre termini otterrò sempre una versione sottocampionata del segnale:

$$\left(\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4}\right) \tag{3.4}$$

$$\left(\frac{1}{2},\frac{1}{2},0,0\right) \qquad \textit{2-term truncation} \tag{3.5}$$

$$(1,0,0,0) \tag{3.6}$$

Al contrario la funzione $f(x) = sincx$ mostra gli artefatti che si generano nel dominio del tempo troncando la serie di Fourier. Infatti la DCT mostra la sequenza tramite l'interferenza delle onde di varie frequenze. Perciò se tronchiamo la serie, in questo caso otteniamo una versione del segnale filtrata attraverso un'analisi passabasso:

$$\left(\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4}\right) \tag{3.7}$$

$$\left(\frac{3}{4},\frac{1}{4},-\frac{1}{4},\frac{1}{4}\right) \qquad \textit{2-term truncation} \tag{3.8}$$

$$(1,0,0,0) \tag{3.9}$$

All'interno dell'esempio 3 sopra riportato, è utile confrontare l'approssimazione del segnale utilizzando i primi due termini della scomposizione perché, anche se con la DWT nel dominio della frequenza risultano un'approssimazione buona, in quello del tempo il segnale sembra avere problemi in quanto mostra dei picchi molto meno pronunciati. Infatti, anche se nel caso dell'approssimazione tramite la DCT, tutti i punti contengono un errore pari a $\frac{1}{4}$, nel caso che utilizza le wavelet il picco non viene mostrato e un errore più grande di $\frac{1}{2}$ è distribuito sui primi due punti. Il risultato è che, mentre nella DCT troveremo artefatti della forma di picchi secondari nei pressi dei contorni (come nella figura 3.14), nell'analisi Wavelet i contorni si distingueranno in modo meno netto e saranno distribuiti su più punti (figura 3.13). Perciò la scelta dei due formati riguardo ai problemi di una compressione eccessiva rimane molto soggettiva.

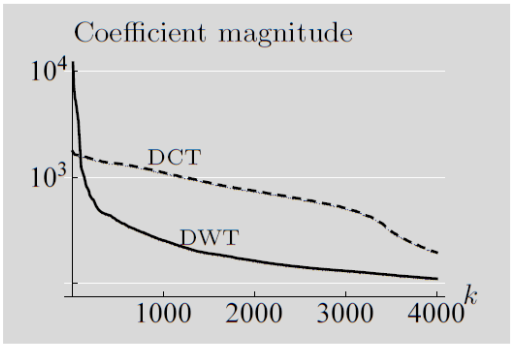
Figura 3.14: Confronto di due immagini compresse nei formati JPEG e JPEG2000. Si può notare nella prima immagine che il mare e il cielo nell'immagine sopra sono più definiti, ma nel campo dell'immagine sopra, nel cielo, e attorno all'aereo si possono vedere facilmente gli artefatti denominati *ringing*, a causa della forma assimilabile ad un "eco" dei contorni.

Un altro vantaggio del formato JPEG2000 dovute alle trasformate e alla libertà di scelta delle funzioni di base wavelet risiede nel rapporto di compressione lossless. Grazie infatti alla particolare base wavelet utilizzata, cioè la wavelet CDF 5/3, il formato JPEG2000, a differenza del predecessore JPEG, può raggiungere risultati molto ottimizzati. Nella tabella seguente sono mostrati i risultati di un test eseguito su un'immagine molto grande (5443x3636 pixel, 16bit di profondità di colore) utilizzando i più comuni formati di compressione lossless:

Tipo di file (e algoritmo utilizzato)	Dimensione del file compresso	Note
JPEG2000 – lossless	61 MB	
JPEG2000 - lossy	21 MB	Perdita di qualità nulla all'occhio umano. Perfetta se il file non necessita ulteriori elaborazioni
TIFF	116 MB	
TIFF con algoritmo LZW	150 MB	L'algoritmo eliminerebbe la ridondanza spaziale, ma è più efficace nei video
PSD	116 MB	Standard di Photoshop

Concludiamo notando che l'estrema flessibilità delle basi wavelet porta questi ottimi risultati nella compressione lossy come in quella lossless grazie alla capacità di concentrare in numero relativamente piccolo di coefficienti la maggior parte dell'informazione. Nell'immagine 3.15 è infatti riportato il grafico che mostra che la maggior parte dei coefficienti wavelet è concentrata nei primi mille termini, e questo è ben visibile anche dalle immagini (c) e (d) in cui la decodifica è ridotta solamente a quelli.

(a) Original 384×576 image x.



(b) Sorted coefficients.

(c) $\breve{x}_{1000}$ with a DCT.

(d) $\breve{x}_{1000}$ with a DWT.

(e) JPEG compression (DCT).

(f) JPEG2000 compression (DWT).

Figura 3.15: Approssimazione e compressione di un'immagine utilizzando la Trasformata Discreta del Coseno su blocchi 8x8, confrontata all'approssimazione non lineare eseguita tramite la Trasformata Discreta Wavelet separabile e ortonormale (esattamente il tipo che abbiamo descritto noi nel paragrafo 2.6. Quest'ultima è stata calcolata tramite i filtri di Daubechies di lunghezza 8. In (e) ed (f) il rapporto di compressione è 0,6 bits per pixel.

Conclusioni

In sintesi questa tesi ha voluto mostrare le caratteristiche peculiari delle trasformate Wavelet che sono risultate essere un interessante sviluppo delle trasformate di Fourier. Attraverso la loro natura localizzata e proporzionale alla scala in analisi, permettono di eseguire un'analisi in multirisoluzione dell'immagine senza richiedere elaborazioni particolarmente complesse: questo dà spazio ad applicazioni che potranno diventare nel corso degli anni sempre più numerose. Le applicazioni nel campo dell'immagazzinamento dei dati in formato digitale possono essere infatti sfruttate anche nel campo di fisica delle particelle, nell'analisi dei modelli di fisica dei sistemi complessi, in campo medico e anche in campi più "umanistici" come la digitalizzazione di documenti storici e letterari.

Ovviamente le applicazioni possono anche essere molto più comuni, come quelle riguardanti la memorizzazione di immagini all'interno del WEB o della televisione, nei quali la velocità di trasmissione dei singoli utenti e la capacità e necessità di decodifica possono avere differenze molto notevoli. L'immagazzinamento nel formato JPEG2000, come abbiamo spiegato, offre infatti importanti soluzioni estremamente flessibili per quanto riguarda la capacità di computazione e di trasmissione del singolo utente. Questo può quindi permettere anche a utenti che vivono in zone più isolate o meno sviluppate del globo la possibilità di accedere alla maggior parte delle risorse presenti nel WEB, o nella televisione, senza la necessità di effettuare adattamenti, spesso complessi, da parte della sorgente del segnale.

Infine è importante sottolineare che questo formato e l'approccio multirisoluzione non sono ancora implementati in buona parte dei software di elaborazione grafica e dei browser. Questo può risultare uno svantaggio, ma dimostra anche che l'analisi multirisoluzione e il JPEG2000 hanno enormi potenzialità poiché fino ad oggi sono stati sviluppati solo da pochi esperti del settore, con studi spesso indirizzati ad applicazioni in campo medico. Al contrario, come è già avvenuto per il formato MP3, l'aumento della sua "popolarità" può spingere grandi aziende e studiosi di questo campo a sviluppare algoritmi sempre più efficienti che sfruttino l'analisi multirisoluzione e le sue potenzialità, e che favoriscano l'implementazione di quest'ultima nei diversi campi d'applicazione.

Bibliografia

[1] Clemens Valens *A Really Friendly Guide to Wavelets.* http://polyvalens.pagesperso-orange.fr/clemens/wavelets/wavelets.html; 1999-2010.

[2] Rafael C. Gonzalez, Richard E. Woods. *Elaborazione delle immagini digitali.* Pearson Prentice Hall, 3 edizione (Ottobre 2008).

[3] Y. Sheng. *The transforms and application handbook.* CRC Press, 1996.

[4] Walter Rudin *Real and Complex Analysis (International Series in Pure and Applied Mathematics).* McGraw-Hill Science/Engineering/Math, 3 edizione, 1 Maggio 1986.

[5] Daubechies, I. *Ten Lectures on Wavelets.* CBMS-NSF regional conference series in applied mathematics, 2 edizione, 1992.

[6] M. Vetterli, J. Kovač evič,V.K. Goyal. *Foundations of Signal Processing.* Cambridge University Press, 1 edizione (Maggio 2014).

[7] M. Vetterli, J. Kovač evič,V.K. Goyal. *Fourier and Wavelet Signal Processing.* Cambridge University Press, 1 edizione (Maggio 2014).

[8] Alan Oppenheim, Ronald Schafer, John Buck. *Discrete-Time Signal Processing.* Pearson Prentice Hall, 2 edizione (1999).

[9] Stephane G. Mallat. *A Theory for Multiresolution Signal Decomposition: The Wavelet Representation.* IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE. VOL. II, NO. 7. JULY 1989.

[10] Stephane G. Mallat. *A Wavelet Tour of Signal Processing.* Academic Press, 1999.

[11] Taubman, David S. *High performance scalable image compression with EBCOT.* IEEE Trans. Image Proc., volume 9, July 2000.

[12] David S. Taubman,Michael Marcellin. *JPEG2000: Image compression fundamentals, standards and practice.* Kluwer Academic Publishers, 2002.