* + - * 1. *Esercitazione n. 3.2: Algoritmo di Mallat*

Partendo dal segnale ECG sperimentale, effettuare i seguenti passi:

* sommare rumore bianco e una sinusoide di ampiezza pari a circa metà dell’ampiezza del segnale ECG e il cui periodo è pari alla lunghezza del segnale ECG, per simulare un artefatto da movimento;
* implementare l’algoritmo di Mallat sia partendo dalla individuazione dei filtri h e g e successivo sottocampionamento del segnale risultante, sia utilizzando la funzione matlab ‘dwt’;
* ricostruire il segnale utilizzando la funzione ‘idwt’ dopo aver eliminato l’artefatto da movimento;
* ricostruire il segnale dopo aver eliminato sia l’artefatto da movimento sia la prima semi-banda (si dovrebbe ottenere una importante riduzione del rumore sul segnale ricostruito).

*Istruzioni matlab per decomporre e ricostruire un segnale:*

* ‘dwt’ decompone e ‘idwt’ ricostruisce un livello per volta; per analizzare più livelli è necessario richiamare la stessa funzione, con l’accorgimento di aggiornare il vettore di ingresso con il vettore ‘approssimazione’ al precedente livello; questa funzione ha il vantaggio di fornire come vettori di uscita il vettore ‘approssimazione’ e il vettore ‘dettagli’ distinti.
* ‘wavedec’ decompone e ‘waverec’ ricostruisce su un numero arbitrario di livelli:
* [xw bv] = wavedec( x, level, 'wavelet'): ‘xw’ è un vettore che contiene tutti i dettagli e l’ultima approssimazione, organizzati in ordine crescente di livello; ‘bv’ contiene la struttura (la lunghezza) dei singoli dettagli e dell’approssimazione;
* x = waverec( xw, bv, 'wavelet'); x è il vettore ricostruito;

Esempio di uso iterativo della dwt (discrete wavelet transform)

n=4; %numero di livelli

ca=s; % segnale da decomporre

Appr=zeros(length(s),n);

Dett=zeros(length(s),n);

for i=1:n

[ca,cd]=dwt(ca,'db2');

Appr(1:length(ca),i)=ca';

Dett(1:length(cd),i)=cd';

figure

subplot(2,1,1), plot(linspace(0,t(end),length(ca)),Appr(1:length(ca),i)),xlabel('t'),title(['approssimazione al livello ',num2str(i)]);

subplot(2,1,2), plot(linspace(0,t(end),length(cd)),Dett(1:length(cd),i)),xlabel('t'),title(['dettaglio al livello ',num2str(i)]);

end



……………………………………………



* 1. **Algoritmo senza decimazione temporale**

Le operazioni di sottocampionamento presenti nell’algoritmo di Mallat eliminano la ridondanza nella rappresentazione del segnale, **ma causano la diminuzione della risoluzione temporale all’aumentare del livello di decomposizione.** In alcune applicazioni può essere importante mantenere la stessa risoluzione temporale su tutti livelli.

Per rispondere a tale esigenza è stato sviluppato un algoritmo di decomposizione senza decimazione nel tempo. La decomposizione del segnale avviene ancora facendo passare il segnale attraverso due filtri g[n] e h[n], rispettivamente passa-alto e passa-basso.

Tuttavia, nell’algoritmo senza decimazione gli stadi di sottocampionamento sono rimossi, mentre le risposte impulsive dei filtri sono interpolate ad ogni livello di decomposizione. **L’interpolazione della risposta impulsiva dei filtri corrisponde alla decimazione dei campioni frequenziali.**

Pertanto, a differenza dell’algoritmo di Mallat, i filtri hanno diversa risposta impulsiva ai diversi livelli di decomposizione, seppur facilmente realizzabili in quanto si possono ottenere mediante interpolazione del filtro allo stadio precedente.

Indicando con *h*[*n*] la generica risposta impulsiva del filtro passa basso e con n la lunghezza del filtro al primo livello, la risposta impulsiva *hk*[*n*] al livello *k-esimo* è ottenuta inserendo () zeri tra ciascuna coppia di coefficiente di *h*[*n*]:



dove k varia tra 1 e il numero massimo di livelli di decomposizione, L è il numero di coefficienti al primo livello del filtro. Per il secondo livello, la lunghezza del filtro è L2 = 2L e così via.

La relazione tempo-frequenza porta alla seguente formula:



dove l’argomento ‘’ di H è stato sostituito con l’argomento ‘’, formalismo utilizzato nelle rappresentazioni discrete. In modo analogo si procede per realizzare il filtro passa-alto.

La risposta impulsiva di h corrisponde alla semi-banda del segnale da decomporre. Quindi al primo livello, essendo k = 1, la risposta in frequenza del filtro *h* è pari a metà della banda del segnale. Quando k = 2 raddoppia la risposta temporale e la risposta in frequenza si dimezza e così di seguito.

Lo schema di funzionamento dell’algoritmo di decimazione in frequenza è riportato in Fig. 24. Il filtro g(n) ha banda complementare ad h(n).

x[n]

g[n]

h[n]

g[n/2]

h[n/2]

g[n/4]

h[n/4]

Coeff\_A3 \_3……

Coeff\_D1

Coeff\_D2

Coeff\_D3

Fig. 24

**Con tale algoritmo è necessario tener conto del diverso ritardo introdotto dai filtri ai vari livelli di decomposizione a causa della diversa durata temporale della risposta impulsiva.** All’aumentare del livello di decomposizione il ritardo temporale introdotto dai filtri aumenta e di ciò bisogna tenere conto nella localizzazione temporale di un evento.

Una delle applicazioni più importanti della decomposizione senza decimazione è nella progettazione di filtri dedicati all’uso in sistemi real-time. A questo proposito si faccia riferimento allo schema a blocchi di Fig. 24. Si può facilmente dimostrare che la risposta in frequenza  che compete al dettaglio del k-esimo livello di decomposizione è data da:



essendo  la risposta in frequenza del filtro g. In particolare: per k = 1, la sequenza di dettaglio (Coeff\_D1) viene filtrata solo con il filtro , senza decimazione in frequenza essendo la prima semi-banda del filtro passa alto; quando k = 2, la sequenza di dettaglio (Coeff\_D2) viene filtrata con  senza decimazione in frequenza (essendo la prima semi-banda del filtro passa-basso al I livello) e con il filtro , che ha subito una prima decimazione in frequenza. Proseguendo con k crescenti si completa la decomposizione.

Quindi la risposta in frequenza  del filtro risultante è il prodotto tra la risposta in frequenza al k-esimo livello e le k-1 risposte in frequenza riferibili alle approssimazioni precedenti (vedi blocchi grigi di Fig. 24.1 per k = 3).

x[n]

g[n]

h[n]

g[n/2]

h[n/2]

g[n/4]

h[n/4]

Coeff\_A3 \_3……

Coeff\_D1

Coeff\_D2

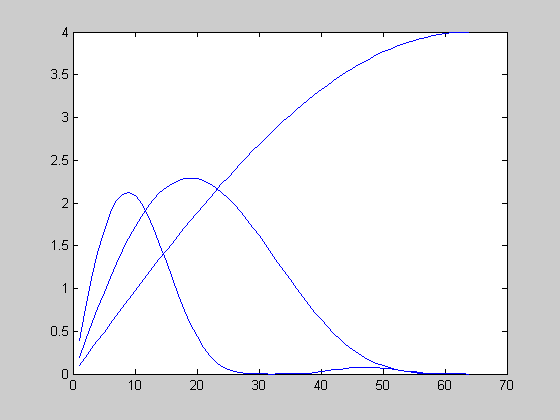
Coeff\_D3

Fig. 24.1

Nel caso interessi la risposta del filtro ‘approssimazione’ al livello k, essa è il prodotto delle risposte in frequenza dei filtri k a tutti i livelli da 1 a k.

Quindi il principale vantaggio di questo metodo è che viene conservata la risoluzione temporale del segnale decomposto. Inoltra per progettare il filtro richiesto si parte dalla massima estensione della banda del segnale da filtrare (che può essere individuata mediante trasformata di fourier) e si ottengono filtri che riflettono le proprietà della decomposizione wavelet (scalabilità, invarianza, ecc.). La risposta impulsiva del filtro è direttamente implementabile su circuiti elettronici, producendo filtri su misura per determinate applicazione..

Tipiche risposte in frequenza per diversi valori di k sono riportate in Fig. 25.



g[n]

h[n]

g[n/2]

Fig. 25