Laboratorio di Telecomunicazioni - a.a. 2010/2011
Lezione n. 9

Trasformata discreta di Fourier

docente L.Verdoliva

In questa lezione affrontiamo il problema dell'elaborazione LTI dei segnali nel dominio della frequenza mediante trasformata discreta di Fourier diretta (DFT) ed inversa (IDFT). Inoltre, condizione di Nyquist, campionamento.

1 La trasformata discreta di Fourier

La trasformata di Fourier di una sequenza causale $x(n)$ di durata finita L è definita come:

$$X(\nu) = \sum_{n=-\infty}^{+\infty} x(n)\, e^{-j2\pi\nu n} = \sum_{n=0}^{L-1} x(n)\, e^{-j2\pi\nu n}$$

e risulta essere una funzione periodica (di periodo 1) e continua nella variabile ν. Per questo motivo l'elaborazione LTI nel dominio della frequenza risulta difficile da implementare numericamente. E' possibile, tuttavia, definire una trasformata discreta di Fourier campionando opportunamente $X(\nu)$ nel periodo, in modo che sia soddisfatta la condizione di Nyquist. Ricordiamo che, se si campiona nel dominio del tempo, questo vincolo richiede che risulti $f_c = \frac{1}{T} \geq 2B$, bisogna cioè campionare ad una frequenza maggiore o uguale della durata frequenziale del segnale, dove con B si è indicata la banda monolatera. Nel caso in cui si effettui il campionamento in frequenza, in modo analogo, deve accadere che l'inverso del passo di campionamento risulti maggiore o uguale della durata temporale del segnale. Se si utilizza un passo di campionamento pari a $1/N$, deve risultare $N \geq L$, cioè bisogna che ci siano un numero di campioni sul periodo maggiore o uguale della durata del segnale. Scegliendo $N = L$, la DFT (Discrete Fourier Transform) risulta essere:

$$X\left(\frac{k}{N}\right) \equiv X_k = \sum_{n=0}^{N-1} x(n)\, e^{-j2\pi nk/N} \qquad k = 0, \cdots, N-1$$

Notate la somiglianza di questa relazione con quella che definisce i coefficienti di Fourier di un segnale periodico, in effetti le equazioni sono identiche a meno di un fattore di scala $(1/N)$. Questo grazie al fatto che un campionamento in frequenza corrisponde ad una periodicizzazione nel dominio del tempo, per cui non stiamo facendo altro che calcolare i coefficienti dello sviluppo in serie di Fourier del segnale periodico di periodo N che ha come generatore proprio $x(n)$.

L'equazione di sintesi (IDFT) è:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X_k \, e^{j2\pi nk/N} \qquad n = 0, \cdots, N-1$$

Per quanto riguarda l'implementazione in Matlab si può utilizzare esattamente lo stesso codice sviluppato per la serie di Fourier, considerando i diversi approcci. Ricordate che quello più efficiente prevede di realizzare un prodotto matriciale.

In realtà in Matlab esiste già un comando per il calcolo della DFT, che risulta computazionalmente molto efficiente perchè implementato mediante algoritmi veloci, noti come FFT (Fast Fourier Transform). Basta allora scrivere il seguente codice dove `x` è il vettore di cui si vuole valutare la DFT su N punti:

```
>> x=boxcar(L);          % genera un impulso rettangolare lungo L
>> X=fft(x,N);
>> v=[0:(1/N):(N-1)/N];
>> subplot(2,1,1); stem(v,abs(X));
```

Notate che è possibile specificare il numero di punti su cui calcolare la DFT. Di default tale valore è proprio pari alla durata del segnale $x(n)$, in caso contrario la sequenza è allungata con $N - L$ zeri (*zero padding*), che corrisponde, di fatto, ad effettuare un sovracampionamento in frequenza. Per comprendere meglio questo fenomeno utilizziamo il comando `fft` per stimare lo spettro $X(\nu)$ del segnale $\mathcal{R}_L(n)$:

```
>> subplot(2,1,2); plot(v,abs(X));
```

Scegliete $L = 8$ e fate variare il valore di N assegnando $8, 16, 32, 64$. Notate come per $N = 8, 16$ non sia possibile visualizzare correttamente lo spettro, nonostante risulti $N \geq L$. Tenete infatti presente che il vincolo di Nyquist richiede che poi venga realizzata un'interpolazione ideale (e non lineare come fa il plot) che in questo caso corrisponde ad interpolare con la funzione di Dirichlet. Per questo motivo è necessario sovracampionare lo spettro, considerando N molto più grande di L.

Notate, infine, come i valori della DFT risultino strettamente dipendenti dal numero di punti su cui si valuta la DFT. Infatti quando si considera N maggiore di L il campionamento è più fitto (ma lo spettro non cambia!). Per questo motivo quando si parla di DFT è sempre conveniente specificare su quanti punti è calcolata (N-DFT).

1.1 Uso della DFT per il filtraggio lineare

In questa sezione ci occuperemo di realizzare in Matlab una procedura alternativa alla convoluzione nel tempo che prevede di moltiplicare le DFT del segnale in ingresso e della risposta impulsiva e poi antitrasformare. Purtroppo, se si realizza il prodotto delle DFT della sequenza di ingresso, X_k, e della risposta impulsiva H_k, a causa della periodicizzazione nel tempo, si ottiene la convoluzione circolare delle sequenze nel dominio del tempo, che in generale non è uguale alla convoluzione lineare, operazione a cui siamo interessati per realizzare il filtraggio nel discreto.

Se supponiamo che la sequenza $x(n)$ abbia durata L, mentre la risposta impulsiva sia lunga M, anche la loro convoluzione, $y(n) = x(n) * h(n)$, avrà durata finita pari a $N = L + M - 1$.

Poiché la N-DFT di $y(n)$ è sufficiente a rappresentare il segnale nel dominio della frequenza (stiamo campionando alla frequenza di Nyquist!), segue che il prodotto della DFT su N punti dell'ingresso e della risposta impulsiva, seguita da una IDFT su N punti fornisce l'uscita corretta $y(n)$. Ciò significa che allungando con zeri le sequenze $x(n)$ e $y(n)$ a N punti e poi realizzando la convoluzione circolare si ottiene lo stesso risultato della convoluzione lineare. Quindi pur di effettuare il riempimento con zeri, la DFT può essere usata per realizzare il filtraggio lineare. Riassumiamo quindi i passi da realizzare:

1. scegliere $N \geq L + M - 1$;
2. calcolare la DFT delle due sequenze su N punti, $X(k)$ e $H(k)$;
3. calcolare il prodotto $Y(k) = X(k)H(k)$;
4. determinare la sequenza filtrata $y(n)$ come IDFT di $Y(k)$.

Questa procedura risulta computazionalmente più efficiente del calcolo diretto della convoluzione. Notate però che in questo modo possiamo realizzare solo il filtraggio di sistemi FIR, dato il vincolo su $h(n)$ che deve avere durata limitata.

Vogliamo determinare attraverso la DFT e la IDFT la risposta del filtro FIR con $x(n) = [1, 2, 3, 4]$ e $h(n) = [2, 1, 2, 1]$, entrambe causali. In questo caso bisogna calcolare la DFT almeno su $N = 4 + 4 - 1 = 7$ punti per garantire la correttezza del risultato:

```
>> x=[1 2 3 4];                    % vettore di ingresso
>> h=[2 1 2 1];                    % risposta impulsiva
>> X=fft(x,7);                     % realizza la DFT su 7 punti
>> H=fft(h,7);
>> Y=X.*H;
>> y=ifft(Y,7)
```

1. Verificate che l'uscita è la stessa che fornisce il comando `y=conv(x,h)`;
2. Ripetete l'operazione di filtraggio in frequenza calcolando la DFT su 4 punti;
3. Scrivete uno script in Matlab per il calcolo della convoluzione circolare di una sequenza e verificate che l'uscita al punto precedente fornisce proprio la convoluzione circolare di $x(n)$ e $h(n)$. (Suggerimento: potete utilizzare il comando `padarray` di Matlab che permette di estendere un vettore periodicamente ai bordi).

1.2 Filtri nel dominio della frequenza

Proviamo adesso a progettare un filtro direttamente nel dominio della frequenza, e consideriamo un filtro passabasso ideale, che ha risposta in frequenza:

$$H(\nu) = \text{rect}(\nu/2B)$$

dove B è la banda monolatera del filtro. Questo filtro, sebbene non possa essere realizzato fisicamente, può però essere simulato al calcolatore col seguente codice:

```
>> x = [4 4 4 4 16 16 16 16 16 16 16];
>> B = 0.4;
>> N = length(x);
>> X = fft(x,N);
>> v = [0:(1/N):(N-1)/N];
>> H = (v <= B) | (v >= 1-B);
>> Y = H.*(X);
>> y = real(ifft(Y));
>> h = real(ifft(H));
>> figure;
>> subplot(2,1,1); stem(y);
>> subplot(2,1,2); stem(fftshift(h));
```

Provate adesso a progettare un filtro passaalto e passabanda, e osservate sia la sequenza filtrata che l'andamento della corrispondente risposta impulsiva.

1.3 Esempi ed esperimenti proposti

a) *Interpretazione della DFT come serie di Fourier.* Provate ad utilizzate il comando `fft` per calcolare i coefficienti della serie di Fourier di un treno di impulsi rettangolari e verificate che vi dà lo stesso risultato della funzione `dfs` dell'esercitazione 5 modificata opportunamente per tenere in conto del fattore di scala.

b) *Elaborazione in frequenza di una riga di un'immagine.* Provate a progettare filtri ideali passa-alto, passa-banda, elimina-banda ed eseguite il codice su una riga di un'immagine.

d) *DFT di un'immagine.* Scriviamo il codice Matlab per realizzare la trasformata di Fourier dell'immagine rettangolo.jpg:

```
>> Y=fft2(double(X));
>> figure(1); imagesc(abs(fftshift(Y))); colormap(gray(256));
>> Z=log(1+abs(fftshift(Y)));
>> figure(2); imagesc(Z); colormap(gray(256));
```

Notate che è stato necessario effettuare il logaritmo dei valori ottenuti per consentire una corretta visualizzazione (la dinamica dei dati è notevolmente cambiata, verificatelo valutando il valore massimo e minimo dell'immagine originale e di quella trasformata). Visualizzate, inoltre, una riga dell'immagine:

```
>> figure(2); plot(abs(fftshift(Y(256,:))));
```

Potevate prevederne l'andamento?

La trasformata di Fourier fornisce un'utile analisi del contenuto frequenziale di un'immagine, provate a calcolare la trasformata delle immagini circuito.jpg e impronta.jpg e cercate di interpretare il risultato.