Laboratorio 7 – Matlab 3.0

Analisi in frequenza di segnali

Questa esercitazione ha come obiettivo l'analisi in frequenza di **segnali acquisiti e generati tramite simulazione**, oppure forniti come **tracce memorizzate in un file**.

Parte 1. Rappresentazione dello spettro di una sinusoide in forma lineare e logaritmica

Si definisca un segnale sinusoidale semplice con frequenza f0, come fatto nelle scorse esercitazioni

```
x=A*sin(2*pi*f0*t);
```

dove t sarà il vettore dei tempi utilizzato per simulare il campionamento **(t=[0:N-1]*1/Fs).** Scegliere Fs adeguata in base alla f0.

Per passare al dominio delle frequenze si utilizza il calcolo della discrete Fourier transform (DFT), realizzato mediante l'algoritmo noto come fast Fourier transform (FFT). A questo scopo MatLab mette a disposizione la function **`fft( )`**, che opera su un vettore di campioni del segnale considerato e **restituisce un vettore complesso**, della stessa dimensione, **contenente i coeffcienti di Fourier del segnale stesso**.

Dato un generico vettore x, i coefficienti di Fourier complessi si ottengono con l'istruzione:

```
X = fft(x);
```

dove

x sarà il vettore dato in ingresso in cui è contenuto il segnale di cui vogliamo calcolare lo spettro

X sarà il vettore restituito in uscita, costituito dai numeri complessi espressi nella forma a + jb che rappresentano i cofficienti di Fourier.

Poichè la misura richiesta è lo spettro di ampiezza del segnale e richiede la presentazione su un grafico dell'andamento del modulo dei coefficienti di Fourier in funzione della frequenza, non sarà sufficiente plottare direttamente il risultato contenuto in X ma saranno necessari i seguenti passi:

1. calcolare il **vettore dei moduli dei coefficienti** di Fourier mediante la funzione **`abs( ).`**
2. normalizzare il vettore ottenuto rispetto al numero di campioni N che lo compongono, poichè il risultato calcolato con l'algoritmo fft non è ancora normalizzato per l'ampiezza della finestra. Definito quindi **`N=length(x)`** sarà sufficiente utilizzare l'istruzione:
   `X = X/N;`
   N.B. Volendo, le istruzioni 1. e 2. possono essere combinate in un'unica istruzione:
   `X=abs(X)/N;`
3. associare un opportuno asse delle frequenze f, da costruire ricordando che la distanza che separa i campioni in frequenza è la granularità in frequenza pari a **`F=1/Tw=1/(Ts*N)=Fs/N`**, dove N è la lunghezza del vettore x, ossia: **`N=length(x).`**
   Tale vettore può per esempio essere creato con l'istruzione: **f=[0:N-1]*Fs/N**

Si crei ora una prima versione del grafico:

```
figure
plot (f, X)
```

Si può osservare che lo spettro così ottenuto è simmetrico rispetto alla frequenza Fs/2 e, data la periodicità degli spettri ottenuti da segnali campionati, la frequenza immagine di f0 si ritrova nella posizione corrispondente a Fs-f0. Lo spettro presenta quindi **due picchi**, corrispondenti alle due righe caratteristiche dello spettro di una sinusoide. In pratica, i **punti di massimo corrispondono agli indici k0 = round(f0/F) e k0im = round((Fs-f 0)/F), quindi alle frequenze k0*F e k0im*F.**

Questa corrispondenza si può verificare sulla traccia visualizzata utilizzando un cursore (**Data ti**) per determinare le coordinate dei punti. Mediante lo stesso cursore, si può anche verificare che le ampiezze dei due picchi spettrali corrispondono approssimativamente a V0/2, dove V0 è l'ampiezza assegnata alla sinusoide.

Data la simmetria, **non c'è bisogno di rappresentare graficamente l'intero spettro**: di norma ne viene rappresentata soltanto la prima parte e di conseguenza la traccia conterrà un solo picco. Si può quindi procedere con una quarta operazione.

4. Dato un generico vettore di N elementi, x(1:N), per considerarne solo metà è sufficiente specificare, il sottoinsieme di indici a cui ci si vuole riferire, in questo caso: `x(1:N/2);`
   Attenzione però che in questo modo, l'indicazione di ampiezza sarebbe poco significativa e per mantenere un'informazione di **ampiezza corretta l'ampiezza va moltiplicata per 2, per tenere conto anche delle componenti immagine che non vengono osservate.**
   `X=2*X(1:N/2);`
5. Si preferisce infine di solito tarare l'asse verticale in modo che l'indicazione corrisponda al valore efficace dell'onda sinusoidale, pertanto i valori vanno ulteriormente divisi per la radice di 2**:** `X=X/sqrt(2);`
   Il grafico che si ottiene ora inserendo le istruzioni:

```
f=f(1:N/2);% questa istruzione è necessaria per coordinare l'asse delle frequenze con la lunghezza di X che ora è stata dimezzata.
figure
plot (f, X)
```

   fornisce così indicazioni **in scala lineare** corrette sia in frequenza (l'unico picco individua la frequenza della sinusoide), sia in ampiezza (il suo valore dà il valore efficace della sinusoide).
6. Per meglio visualizzare componenti di ridotta ampiezza nelle misure di spettro i valori di ampiezza sono in molti casi presentati su una scala logaritmica sull'asse verticale. Questo grafico è ottenibile attraverso l'istruzione: `plot(f,20*log10(X));`
   dove X sarà il vettore spettro espresso in termini di valore efficace e fornisce indicazioni in decibel rispetto ad un riferimento di 1 V efficace, per cui si utilizza l'unità dBV.

Parte 2. Osservazione della dispersione spettrale

Si provi ora a visualizzare lo spettro di sinusoidi con **almeno due** differenti valori di f0, scegliendo:

- ➔ Un valore per cui f0 sia multiplo di F (esiste k0 per cui avrò f0=k0*F)
- ➔ Un valore per cui f0 non sia un multiplo di F (non esiste un k intero per cui f0=k0*F)

Osservare in che situazione compare l'effetto della **dispersione spettrale**, utilizzando la scala logaritmica appena introdotta. Questa mette in particolare evidenza il fenomeno della dispersione spettrale, che può indurre un peggioramento della qualità della misura soprattutto quando lo spettro del segnale sarà formato da più componenti (vedi parte 4).

Parte 3. Utilizzo delle finestre per analisi spettrale

Finora nel calcolo dello spettro è stata implicitamente utilizzata una "finestra rettangolare". Il vettore dei campioni corrisponde cioè ad un'osservazione del segnale che è stata troncata entro un intervallo di durata finita.

In generale, la finestra è una funzione che permette di pesare diversamente i campioni acquisiti, secondo la loro posizione nel vettore e, di solito, il peso dei campioni posti agli estremi tende a 0.

Tra quelle maggiormente utilizzate per ridurre la dispersione spettrale vi è la finestra di **Hann**, la cui lunghezza deve coincidere con la lunghezza del vettore di campioni considerato.

Digitando nella command window "**`help window`**" Matlab vi fornisce un elenco completo delle finestre che possono essere create.

Con riferimento ad un generico vettore x, la cui lunghezza sarà pari a N, i campioni della finestra si possono generare tramite il comando MatLab:

```
N=length(x);
```

```
w=hann(N);
```

(volendo utilizzare altre finestre il procedimento sarà lo stesso cambierà solo l'istruzione per creare w, per esempio volendo utilizzare una finestra Flat Top, utilizzeremo l'istruzione:

```
w=flattopwin(N);
```
)

La DFT viene poi calcolata sui campioni del segnale **pesati per la funzione di finestratura ed opportunamente scalati, ovvero normalizzati per la somma dei pesi dei campioni della funzione finestra.** Questo corrisponde al passaggio 2) svolto nella parte 1, dove assumendo implicitamente una finestra rettangolare, il vettore X veniva normalizzato per il valore N che corrisponde infatti alla somma dei pesi della finestra essendo la finestra standard di ampiezza 1. In questo caso in cui i campioni hanno valori diversi da 1, tale somma può essere calcolata tramite la funzione **sum(w).**

Si ha così la sequenza di istruzioni MatLab:

```
xw = x.*w;
```

```
Y = fft(xw)/sum(w);
```

oppure la singola istruzione:

```
Y =  fft(x.*w)/sum(w);
```

Va notato che si utilizza l'operatore .*, dove il punto indica l'operazione di prodotto scalare tra elementi corrispondenti dei due vettori, che devono ovviamente avere uguale lunghezza.

ATTENZIONE! Si deve verificare che:

- ➔ *i vettori x e w, oltre ad avere lo stesso numero di elementi, siano anche definiti* ***entrambi come vettori riga, oppure entrambi come vettori colonna****. Se così non è, nel prodotto .* uno dei due vettori andrà trasposto, ad esempio scrivendo: x.*w';*
- ➔ *la FFT* ***non sia più scalata per il numero di elementi del vettore, ma solo per la somma dei valori della finestra (sum(w)),*** *come nelle istruzioni MatLab sopra riportate.*

Confrontare gli spettri ottenuti senza finestra, con la finestra di Hanning e con la Flat Top sovrapponendoli nella stessa figura (usare il comando hold on prima di inserire il secondo plot nella stessa figure).

Finestra	WCSL [dB]	attenuazione minima lobi laterali [dB]	larghezza lobo princ. [bin]	banda a -6dB [bin]	ENBW [bin]
uniforme	3.92	13	2	1.21	1
Hann (*Hanning*)	1.42	32	4	2	1.5
Blackman-Harris	1.13	71	6	2.27	1.71
flat-top	< 0.01	93	10	4.58	3.77

Parte 4. Sinusoidi multiple

Si provi a creare **un segnale costituito da più sinusoidi a frequenze differenti e non armoniche** (non multiple tra loro). *(Selezionare del tutto liberamente la composizione. Un esempio a pure titolo di esempio potrebbe essere un segnale somma di tre sinuosidi rispettivamente con f1=5; f2=34; f3=127;)*

1. Scegliere l'**intervallo di osservazione** in modo che corrisponda a molti periodi *(almeno 20)* delle sinusoidi e calcolarne lo spettro sia con finestra di Hanning, sia senza.
2. Diminuire l'**intervallo di osservazione** ed osservare come variano gli spettri; al fine di ottenere un confronto efficace mostrare sullo stesso grafico i nuovi spettri e quelli ottenuti con l'intervallo di osservazione lungo *(Per esempio confrontare gli spettri ottenuti con 20, 10, 5 e 2 periodi)*

Lo spettro di un segnale è utile non solo per un'analisi qualitativa ma anche quantitativa e fornisce svariate informazioni, in termini sia delle **frequenze che delle ampiezze delle varie componenti del segnale tramite l'individuazione dei picchi dello spettro con maggiore ampiezza.**

Come si può osservare dal confronto tra le diverse tracce ottenute, la possibilità di estrarre informazioni quantitative dallo spettro dipende anche da una scelta adeguata dell'intervallo di osservazione e della finestra utilizzata.

Parte 5. Analisi di registrazioni reali

Questa parte dell'esercitazione riguarda un esempio di analisi di acquisizioni reali, i cui campioni sono contenuti **in file .mat**.

Le tracce contenute nei file sono un segnale elettrocardiografico (ECG) acquisito da un elettrocardiografo portatile ed un segnale fotopletismografico (photopletismography, PPG) ottenuto da un pul siossimetro. Entrambi contengono informazioni relative alla frequenza cardiaca del soggetto (ogni "onda" del segnale corrisponde ad un battito cardiaco) e, ovviamente, il segnale ECG contiene anche un maggior grado di dettaglio.

I due segnali sono stati acquisiti simultaneamente dallo stesso soggetto, utilizzando la stessa frequenza di campionamento di 125 Hz.

Potete trovare i segnali ECG.mat e ppg.mat nella cartella relativa al LAB7 nella sezione esercitazioni Matlab su moodle. I file contengono ciascuno 1500 campioni, corrispondenti quindi a 12 s di registrazione. Per ciascun segnale fornito, il corrispondente file nomefile.mat va caricato nel Workspace MatLab usando il comando load.

```
load('ECG.mat');
```

```
load('PPG.mat');
```

5.1 Osservazione nel tempo

Osservando in un plot l'andamento dei due segnali, ci si può rendere conto che il loro valore medio non è nullo, ossia che entrambi comprendono una componente continua, o lentamente variabile, che però non interessa nell'analisi che segue.

Il modo più semplice di escludere questa componente dall'analisi è sottrarre al vettore dei campioni segnale il proprio valore medio. Per un generico vettore x, questo si ottiene con l'istruzione:

```
x = x - mean(x);
```

che fa uso della function **`mean( )`** di MatLab.

5.2 Analisi in frequenza

Per passare dall'osservazione nel dominio del tempo all'analisi nel dominio delle frequenze si deve calcolare la discrete Fourier transform (DFT) dei dati acquisiti, seguendo le avvertenze descritte nei paragrafi precedenti.

ATTENZIONE!!

1. Si consiglia di utilizzare la finestra di Hanning (data N=length(x) la finestra sarà w=Hann(N) e il segnale finestrato x.*w) e quindi i coefficienti di Fourier ottenuti vanno normalizzati, come indicato nei paragrafi precedenti, per la somma di tutti i coefficienti della finestra pari a sum(w)
2. Si ricorda che data la simmetria va rappresentata solo metà dello spettro in frequenza che va da 0 a Fs, raddoppiando però le ampiezze per tenere conto della parte immagine che si è omessa (X=2*X(1:N/2) e ovviamente anche il vettore delle frequenze sarà f=[0:N/2-1]*Fs/N)

3. Si può decidere la lunghezza dell'intervallo da considerare, che non deve necessariamente coincidere con l'intera traccia registrata.
4. Per l'analisi di segnali biomedici si utilizza in genere una scala lineare anche per le ampiezze.

Come noto, **la frequenza cardiaca di un soggetto normale varia circa tra 50 e 200 battiti al minuto (BPM, beats per minute), che corrispondono ad una frequenza fondamentale compresa tra 0.8 e 3.5 Hz.**

Utilizzando un grafico ottenuto sovrapponendo in una figure i due plot, si confrontino gli spettri ottenuti per il segnale ECG e PPG, verificando se le indicazioni ottenibili riguardo alla frequenza cardiaca sono in accordo.

5.3 Ricerca delle componenti periodiche del segnale

Quando si analizzano tratti di durata limitata, i segnali ECG e PPG si possono ritenere periodici o più esattamente pseudoperiodici, a causa delle variazioni che si verificano nel medio-lungo periodo. In particolare, il battito cardiaco è caratterizzato da una **variabilità intrinseca (heart rate variability, HRV),** il cui studio può a sua volta fornire informazioni diagnostiche interessanti.

La determinazione delle principali componenti periodiche del segnale, e più nello specifico della componente relativa alla frequenza cardiaca che come detto sopra rientrerà fra gli 0.8 e i 3.5 Hz, si può fare seguendo tre metodi:

METODO 1:

CURSORI: attivando tra le icone presenti sull'immagine in alto a destra "Data tips" avete la possibilità di inserire dei cursori che permettono di determinare le coordinate di ampiezza e frequenza dei picchi spettrali. Le componenti devono essere armoniche, quindi le frequenze devono risultare multiple intere della frequenza fondamentale, nei limiti dell'accuratezza della misura. Identificando manualmente il massimo tra i picchi rilevati nell'intervallo compreso tra 0.8 e 3.5 Hz si potranno avere a disposizione frequenza e ampiezza della componente fondamentale, corrispondente alla frequenza cardiaca. E' un metodo semplice e rapido se si vuole fare un singolo check su un'immagine, ma sicuramente non è ottimale in quanto non automatizzato.

METODO 2:

MODALITA' FINDPEAKS BASE ASSOCIATA AD UN ALGORITMO CHE RESTRINGA LA SELEZIONE:

1. Calcolare gli spettri dei segnali ECG (XECGw) e PPG (XPPGw) - senza media, finestrati, solo la prima metà- e plottarli nello stesso grafico.

2. Calcoliamo tutti i picchi del segnale con la funzione MatLab **`findpeaks( )`** nel modo più semplice, cioè fornendo in ingresso solo il vettore data, contenente i dati sui quali si vuole eseguire la ricerca dei massimi locali (ristretti alle prime N/2 componenti per simmetria), l'istruzione:
`[pks,locs] = findpeaks(data);`
restituisce in uscita due vettori, pks contenente l'ampiezza dei picchi e locs che contiene gli indici relativi alla loro posizione.

ATTENZIONE

La ricerca dei picchi riguarda il modulo dello spettro e agirà quindi sul vettore passato come data (sulle N componenti se viene passato data intero, su N/2 componenti se è già stata operata l'operazione di considerare metà vettore per simmetria)

*Per conoscere le frequenze corrispondenti è necessario convertire l'indice numerico fornito da findpeaks( ) in frequenza, moltiplicandolo per il valore F. Va ricordato che gli indici dei vettori in MatLab iniziano dal valore 1, mentre nel vettore dei coefficienti di Fourier il primo elemento fa riferimento alla frequenza 0. La conversione da indici (locs) a valori di frequenza si ottiene quindi calcolando (locs-1)*F.*

Per plottare i valori dei picchi sul grafico è sufficiente utilizzare la seguente istruzione:

```
plot(f(locs),peaks,'ro
```

3. Abbiamo tantissimi picchi.. Come facciamo a sapere qual è quello corretto per identificare la frequenza cardiaca?

- Se il segnale è «SEMPLICE» e contiene pochi picchi come il PPG, il picco di ampiezza massima è quello dell'HR.

```
[~,indice_max] = max(pks);
indice_hr_ppg = locs1(indice_max);
HR_ppg = (indice_hr_ppg-1)*F;
```

- Se il segnale è complesso e ci sono tante armoniche (come per l'ECG), non è detto che il valore massimo corrisponda alla freq. Cardiaca. Perciò dovremo cercare il massimo nel range di valori accettabili (cioè tra 0.8 e 3.5 Hz).

```
f_locs = (locs-1)*F;
indici_range = find(f_locs>0.8 & f_locs<3.5);
[~,indice_max] = max(pks(indici_range));
indice_hr_ecg=locs(indici_range(indice_max));
HR_ecg = (indice_hr_ecg-1)*F;
```

METODO 3:

MODALITA' FINDPEAKS CON SOGLIA ASSOCIATA AD UN ALGORITMO CHE RESTRINGA LA SELEZIONE

Questo metodo ripercorre esattamente il metodo 2 con un accorgimento nell'utilizzo della funzione findpeaks che ha come obiettivo quello di ridurre un po' il numero di picchi considerati, scartando quelli con ampiezza troppo bassa che sono sicuramente picchi di rumore.

La funzione findpeaks utilizzata con solo il vettore in ingresso esegue una ricerca di massimi relativi che porta ad individuare numerosi picchi scarsamente significativi: un metodo più raffinato per limitare i picchi a quelli significativi consiste nel passare alla funzione findpeaks sia il vettore dei valori dello spettro, sia il vettore f dei valori di frequenza corrispondenti, che un valore soglia di ampiezza **th**, per fare in modo che eventuali picchi inferiori alla soglia siano ignorati. (Il valore di tale soglia è scelto dall'osservatore tramite un'osservazione preliminare del segnale nel tempo). In questo caso l'istruzione prende la forma:

```
[pks,locs] = findpeaks(data,'MinPeakHeight', th);
```

dove alla generica denominazione data va sostituito il vettore dei valori calcolati dello spettro.

Operativamente procedere quindi con i seguenti passaggi:

1. Osservando il grafico selezionare un valore di soglia (th) che consenta di scartare i picchi di rumore, selezionando solo quelli più grandi (ad es. th =8).

2. Identificare tutti i picchi del segnale che hanno ampiezza superiore alla soglia.

```
th = 8;
[pks,locs] = findpeaks(abs(X_ECGw), 'MinPeakHeight', th);
[pks1,locs1] = findpeaks(abs(X_PPGw), 'MinPeakHeight', th);
```

3. Plottare i picchi trovati.

4. Procedere come visto al punto 3 del Metodo 2 per calcolare la frequenza cardiaca.

Date le modalità di acquisizione descritte, si noti che:

1) la **frequenza fondamentale identificata automaticamente tramite i metodi sopra, che indica la frequenza cardiaca, deve essere la stessa nella traccia ECG e nella traccia PPG**. Moltiplicando il valore in Hz per 60 si ottiene il numero di battiti cardiaci per minuto (BPM).

2) la **composizione spettrale dei due segnali, invece, è diversa** ed il segnale ECG contiene un maggior numero di componenti.