

Laboratorio di Telecomunicazioni - a.a. 2010/2011

Lezione n. 1

Rappresentazione dei segnali a tempo discreto

L. Verdoliva

In questa prima lezione affrontiamo il problema della generazione e rappresentazione dei segnali a tempo discreto.

1 Generazione di segnali a tempo discreto

In Matlab, un segnale a tempo discreto (o sequenza) $x(n)$ di durata finita può essere rappresentato da una *coppia* di vettori riga (o colonna):

- un vettore **n** che rappresenta gli istanti di tempo;
- un vettore **x** che rappresenta i valori assunti dal segnale.

Ad esempio, il segnale $x(n) = 2\delta(n+3) + \delta(n+2) - \delta(n+1) + \delta(n-1) + 4\delta(n-2) + 3\delta(n-3)$ viene definito nel modo seguente:

```
>> n = [-3,-2,-1,0,1,2,3];  
>> x = [2,1,-1,0,1,4,3];
```

Ricordiamo che il punto e virgola “;” al termine del comando serve ad impedire che Matlab stampi a video l’output del comando impartito, si osservi pure che

```
>> n = [-3,-2,-1,0,1,2,3];
```

può essere abbreviato con

```
>> n = [-3:3];
```

che genera tutti i numeri fra -3 e 3 (inclusi) con passo 1. Se il segnale è causale, cioè è definito a partire da $n = 0$, il vettore dei tempi **n** assume i valori $0, 1, 2, \dots, N-1$, con N è la lunghezza del vettore **x**. In Matlab l’asse dei tempi può allora essere definito nel seguente modo:

```
>> n = [0:length(x)-1];
```

dove **length(x)** permette di ottenere la lunghezza N di un vettore.

Attenzione: in Matlab, a differenza di altri linguaggi di programmazione (es. C, C++), gli indici dei vettori vanno da 1 in poi: questo va tenuto presente quando si associano i valori di **n** agli indici del vettore. Ad esempio, se abbiamo definito i tempi a partire da $n = 0$, **x(i)** rappresenta il campione all’istante $n = i - 1$, e non quello all’istante $n = i$.

1.1 Rappresentazione grafica di segnali a tempo discreto

In Matlab, un segnale a tempo discreto generato come descritto in precedenza può essere rappresentato graficamente con il comando `stem` :

```
>> stem(n,x);
```

È possibile poi modificare e/o aggiungere elementi al grafico utilizzando i comandi `xlabel`, `ylabel`, `title`, `axis`. Ad esempio, la coppia di comandi

```
>> stem(n,x,'filled'); axis([-4 4 -2 5]);
```

genera una rappresentazione più chiara del segnale. Per dettagli sull'uso di questi ed altri comandi, si consulti l'help in linea di Matlab.

1.2 Esempi ed esperimenti proposti

- a) *Finestra rettangolare a tempo discreto.* Generate e diagrammate una finestra rettangolare a tempo discreto $\mathcal{R}_N(n)$ per $-20 \leq n \leq 20$ e $N = 10$.

È possibile sicuramente scrivere un codice per questo esempio basato su un ciclo `for`, ma sarebbe una soluzione inefficiente. Possiamo pensare ad una soluzione che sfrutta le operazioni logiche su matrici. Ad esempio, possiamo scrivere:

```
>> n = [-20:20];  
>> x = [(n >= 0) & (n <= 9)];  
>> stem(n,x);
```

La prima istruzione genera un vettore di valori interi (tempi) da -20 a 20 . La seconda genera un vettore che vale 1 nelle posizioni in cui n è compreso tra 0 e 9 (estremi inclusi), zero altrimenti. La terza disegna il segnale generato. Per ottenere una figura più leggibile conviene definire gli estremi della figura con il comando `axis`, ad esempio come

```
axis([min(n)-1 max(n)+1 min(x)-1 max(x)+1]);
```

Potete provare a trasformare questo esempio in una `function` per generare una qualsiasi finestra rettangolare tra `n1` ed `n2` e di lunghezza `N`:

```
function [x,n] = rett(n1,n2,N);  
n = [n1:n2];  
x = [(n >= 0) & (n <= (N-1))];
```

La funzione restituisce `x` ed `n`, e va memorizzata in un *M-file* con il nome `rett.m`; essa potrà essere richiamata dalla linea di comando nel seguente modo:

```
>> [x,n] = rett(-20,20,10);  
>> stem(n,x)
```

b) *Finestra triangolare a tempo discreto.* Generate e diagrammate una finestra triangolare a tempo discreto $\mathcal{B}_{2N}(n) = \left[1 - \frac{|n-N|}{N}\right] \mathcal{R}_{2N}(n)$ per $-20 \leq n \leq 20$ e $N = 5$. Costruite una funzione **triang** simile a **rett** per la generazione di una finestra triangolare di lunghezza arbitraria.

c) *Impulso e gradino a tempo discreto.* Generate e diagrammate un impulso ideale a tempo discreto $\delta(n)$ e un gradino unitario a tempo discreto $u(n)$ per $-20 \leq n \leq 20$ (scrivete il codice evitando i cicli **for**).

Costruite poi una funzione **delta** per la generazione di un impulso definito in un generico istante n_0 nell'intervallo $n_1 \leq n_0 \leq n_2$; la funzione potrebbe avere il seguente prototipo:

```
function [x,n] = delta(n0,n1,n2)
```

Allo stesso modo create una funzione **grad** per la generazione del gradino unitario $u(n-n_0)$ nell'intervallo $n_1 \leq n_0 \leq n_2$.

d) *Esponenziale a tempo discreto.* Generate e diagrammate un esponenziale monolatero a tempo discreto $x(n) = a^n u(n)$ per $-20 \leq n \leq 20$ e per vari valori di a (positivi, negativi, maggiori o minori di 1 in modulo).

e) *Sinusoide a tempo discreto.* Generate e diagrammate la sinusoide a tempo discreto $x(n) = 3 \cos(0.1\pi n + \pi/3)$ nell'intervallo $0 \leq n \leq 10$.

- Verificate sperimentalmente che la sinusoide a tempo discreto è periodica se la frequenza numerica ν_0 è un numero razionale. A tal fine:
 - generate e diagrammate $\cos(0.3\pi n)$, $-20 \leq n \leq 20$. Qual è la frequenza ν_0 di tale sinusoide? Tale sequenza è periodica? Se sì, qual è il suo periodo?
 - generate e diagrammate $\cos(0.3n)$, $-20 \leq n \leq 20$. Qual è la frequenza ν_0 di tale sinusoide? Tale sequenza è periodica? Se sì, qual è il suo periodo?
- Verificate sperimentalmente che due sinusoidi a tempo discreto aventi frequenze che differiscono per un numero intero coincidono, diagrammando $\cos(2\pi 0.3n)$ e $\cos(2\pi 1.3n)$.
- Verificate che incrementando la frequenza numerica ν_0 da 0 ad $1/2$ la rapidità di variazione aumenta, per poi diminuire quando la frequenza passa da $1/2$ ad 1. A tal fine rappresentate graficamente la sinusoide per $\nu_0 = 0, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, \frac{7}{8}, \frac{15}{16}, 1$.

f) *Sequenze periodiche.* Una sequenza $x_p(n)$ è periodica se $x_p(n) = x_p(n+N)$, $\forall n$. Il più piccolo intero n che verifica tale relazione è detto *periodo fondamentale*. Indicando con $x(n)$ il generatore della sequenza, cioè $x_p(n) = \text{rep}_N[x(n)]$, per generare P periodi della sequenza si può replicare $x(n)$ P volte con il seguente comando:

```
>> xp = [x, x, ..., x];
```

Un approccio più elegante è quello di usare la capacità di indicizzazione del Matlab: si crea prima una matrice contenente P righe con i valori di $x(n)$ per poi concatenarle in un vettore usando l'operatore colonna (:). Per esempio se il generatore è rappresentato dal vettore $x = [5, 4, 3, 2, 1]$, definito a partire da $n = 0$, e se si vogliono visualizzare 4 periodi del segnale nell'intervallo $n = [-10 : 9]$, bisogna scrivere il seguente codice:

```
>> n=[-10:9]; x=[5, 4, 3, 2, 1];  
>> xp = x'*ones(1,4);  
>> xp=(xp(:))';  
>> stem(n,xp); title('sequenza periodica');  
>> xlabel('n'); ylabel('xp(n)')
```

Gli ultimi due comandi servono ad etichettare l'asse delle ascisse e quello delle ordinate.

Generate e diagrammate il segnale periodico $x_p(n) = \text{rep}_8[\mathcal{R}_4(n)]$ (sequenza di impulsi rettangolari) per $0 \leq n < 40$.