Relazione di Elaborazione dei Segnali

Politecnico di Torino AA 2018-2019

Valerio Casalino
233808

Indice

Capitolo 1

Introduzione

1.1 Obiettivi e traguardi

L'obiettivo delle esercitazioni è quello di mettere in pratica, osservare e verificare il livello di apprendimento della materia appoggiandoci sull'ambiente MATLAB [Website link].

1.2 Materiale e documentazione disponibile

Abbiamo a disposizione per lo svolgimento delle esercitazioni, oltre che alle conoscenze pregresse, anche il seguente materiale:

- Documentazione interna di MATLAB, attraverso i comandi *doc* e *help*.
- La sezione dedicata su StackOverflow: `https://stackoverflow.com/questions/tagged/matlab`.
- La community di MATLAB: `https://it.mathworks.com/matlabcentral/?s_tid=gn_mlc`.

1.3 Note

La relazione, come i codici sorgente delle esercitazioni, sono disponibili su GitHub, all'indirizzo `http://bit.ly/vcasalino-github-tes`.

Capitolo 2

LAB1: Convoluzione, mutua correlazione e stima del ritardo

2.1 Convoluzione

> *In matematica, in particolare nell'analisi funzionale, la convoluzione è un'operazione tra due funzioni di una variabile che consiste nell'integrare il prodotto tra la prima e la seconda traslata di un certo valore.*

-Wikipedia.

2.1.1 Convoluzione lineare e discreta

L'operazione di convoluzione tra funzioni continue è definita in tale modo:

$$f \circledast g = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau = \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau \tag{2.1}$$

La convoluzione discreta, invece, è definita come:

$$\sum_{m=-\infty}^{\infty} f[m]g[n-m] = \sum_{m=-\infty}^{\infty} f[n-m]g[m] \tag{2.2}$$

2.1.2 Convoluzione circolare

Data una funzione x_T di periodo T, la sua convoluzione con una funzione h è ancora periodica, si dice convoluzione circolare e si calcola nel seguente modo:

$$(x_T \circledast h)(t) = \int_{-\infty}^{\infty} h(\tau) \cdot x_T(t-\tau)d\tau = \int_{t_0}^{t_0+T} h_T(\tau) \cdot x_T(t-\tau)d\tau \tag{2.3}$$

Dove t_0 è arbitrario e h_T è espresso come:

$$h_T(t) = \sum_{k=-\infty}^{\infty} h(t-kT) \tag{2.4}$$

Data una funzione g_N periodica con periodo N, la sua convoluzione discreta è calcolata come:

$$(f \circledast g_N)[n] = \sum_{m=0}^{N-1} f[m]g[(n-m)_{modN}] \tag{2.5}$$

2.2 Esercizio 1: Convoluzione lineare

Il testo dell'esercizio chiede di eseguire la convoluzione tra le seguenti funzioni:

$$x(n) = \begin{cases} sin(\frac{\pi n}{5}) & \text{se } 0 \leq n \leq 4 \\ 0 & \text{altrove} \end{cases} \tag{2.6}$$

$$y(n) = \begin{cases} 1 & \text{se } 0 \leq n \leq 2 \\ 0 & altrove \end{cases} \tag{2.7}$$

Senza l'ausilio del comando di libreria di MATLAB: *conv()*. Per farlo è necessario applicare loro la formula per la convoluzione discreta (2.1.1) "manualmente". Procediamo quindi alla scrittura del codice per passi:

1. Creazione dei due vettori sui quali applicare la convoluzione. (x ed y).
2. Definizione della lunghezza finale che deve avere il vettore finale e adattamento del vettore x per la lunghezza ricercata.
3. Iterazione con doppio ciclo *for* per ottenere il risultato.

```
>> L1E1

z =

    0.5878    1.5388    2.4899    2.4899    1.5388    0.5878


ans =

         0         0         0    0.5878    1.5388    2.4899    2.4899    1.5388    0.5878         0         0
```

Figura 2.1: Risultato del codice (z), confrontato con la funzione di libreria *conv()*.

I risultati in figura 2.1 differiscono per dimensione solo per l'implementazione della funzione di MATLAB, ma i valori non nulli sono gli stessi.

2.3 Esercizio 2: Convoluzione circolare

Prendendo sempre in considerazione la funzione 2.6 e la 2.7, l'esercizio è similare al precedente e prevede che venga calcolata la convoluzione circolare tra i due segnali senza ricorrere alla funzione di libreria *cconv()*.

Per fare ciò è necessario allocare e dimensionare adeguatamente un vettore z di zeri di dimensione a, che è il massimo tra la lunghezza di x e y. Ciò dipende dal fatto che la convoluzione circolare contiene un numero di campioni pari a tale valore. Fatto ciò è possibile procedere al calcolo in maniera iterativa della convoluzione circolare fra i due segnali riportando gli indici in base 0 e utilizzando la funzione libreria *mod()*.

```
z =

    1.5388    1.1756    1.5388    2.4899    2.4899


ans =

    1.5388    1.1756    1.5388    2.4899    2.4899
```

Figura 2.2: Output dell'esercizio 2.

Il risultato, ottenuto tramite il processo iterativo del codice, che traduce in linguaggio MATLAB la 2.5, fornisce gli stessi valori della funzione di libreria, come si vede in figura 2.2.

2.4 Esercizio 3: Mutua correlazione e stima del ritardo

> Un segnale *x(n)* di durata N campioni viene irradiato periodicamente dall'antenna di un trasmettitore. Un ricevitore mobile (che conosce il segnale *x(n)*) riceve una versione rumorosa e ritardata del segnale trasmesso.

$$r(n) = x(nD) + g(n) \tag{2.8}$$

Dove D è un valore di ritardo (intero) e *g(n)* è un segnale di rumore additivo gaussiano bianco con varianza σ^2.

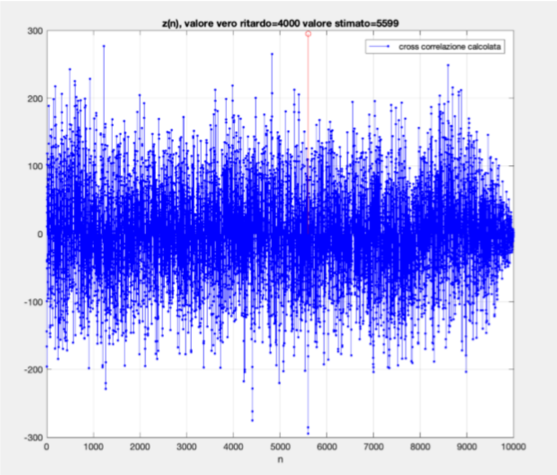
Si procede all'implementazione in MATLAB della funzione *mychannel()*:

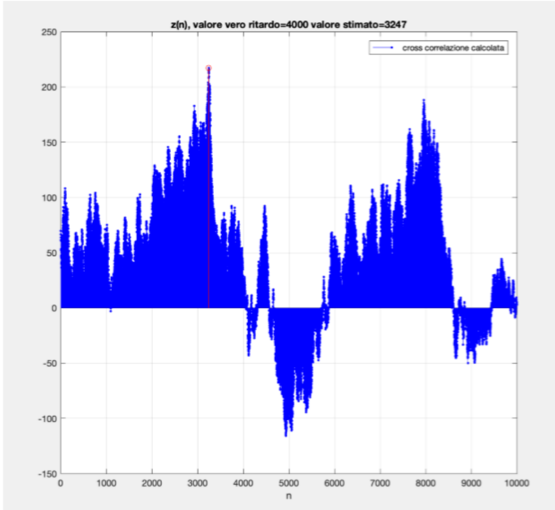
```
function [out1, out2] = mychannel(x, D, sigma)
    samples = length(x);
    shifted_data = delayseq(x, D);
    WN = sqrt(sigma).*randn(1, samples);
    out1 = shifted_data + WN;
    out2 = shifted_data;
end
```

Figura 2.3: Implementazione della funzione mychannel.

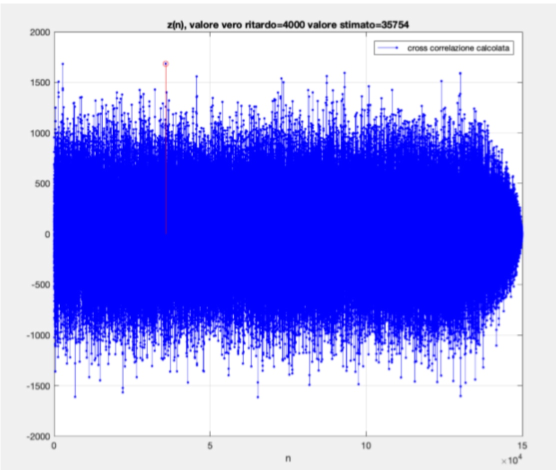
In seguito è possibile produrre un grafico della stima del ritardo D per diversi valori di N e σ^2 sia nel caso in cui *configBit* (presente in Esercitazione13.m) corrisponde a 0 e 1.

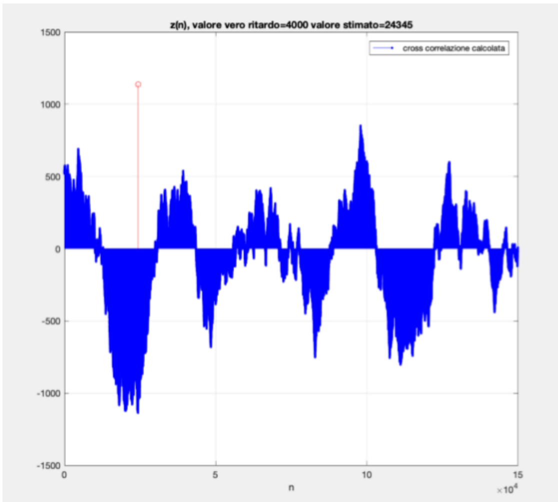
$N = 1000; \sigma^2 = 5$





$N = 15000; \sigma^2 = 10$





Capitolo 3

LAB2: Discrete Fourier Transform

3.1 Esercizio 1: Convoluzione e sistemi LTI

La convoluzione tra due segnali è equivalente al loro prodotto nel dominio della frequenza. Possiamo riscrivere la funzione di convoluzione dell'esercizio precedente sfruttando questa relazione matematica:

$$z(n) = x(n) \circledast y(n) \rightarrow z(n) = F^{-1}(F[x(n)](f) \cdot F[y(n)](f)) \tag{3.1}$$

3.2 Implementazione DFT

In MATLAB è possibile implementare le DFT e le IDFT come segue:

$$x_{out}[k] = \begin{cases} \frac{1}{N} \sum_{n=0}^{N-1} x_{in}[n] e^{j2\pi kn/N} & k = 0, ..., N-1 \\ \sum_{n=0}^{N-1} x_{in}[n] e^{-j2\pi kn/N} & k = 0, ..., N-1 \end{cases} \tag{3.2}$$

dove N è la lunghezza del segnale $x_{in}[n]$.

```
% DFT
X = zeros(1, N);

for i = 0:(N - 1)
    for j = 0:(N - 1)
        X(i + 1) =  X(i + 1) + (x(j + 1) * exp(-(2*pi*j*1i*i/N)));
    end
end
```

Figura 3.1: Implementazione DFT.

```
% IDFT
xi = zeros(1, N);

for i = 0:(N - 1)
    for j = 0:(N - 1)
        xi(i + 1) =  xi(i + 1) + (1/N)*(X(j + 1) * ...
            exp(2*pi*j*1i*i/N));
    end
end
```

Figura 3.2: Implementazione IDFT.

Le funzioni implementate in questo modo risultano essere equivalenti alle funzioni di libreria *fft()* e *ifft()* implementate in MATLAB, come si vede in figura 3.3.

```
X =

   3.0777 + 0.0000i  -1.1756 + 0.0000i  -0.3633 - 0.0000i  -0.3633 - 0.0000i  -1.1756 - 0.0000i

ans =

   3.0777 + 0.0000i  -1.1756 + 0.0000i  -0.3633 - 0.0000i  -0.3633 + 0.0000i  -1.1756 - 0.0000i

xi =

   0.0000 - 0.0000i   0.5878 - 0.0000i   0.9511 - 0.0000i   0.9511 + 0.0000i   0.5878 + 0.0000i

ans =

   0.0000 - 0.0000i   0.5878 - 0.0000i   0.9511 + 0.0000i   0.9511 + 0.0000i   0.5878 + 0.0000i
```

Figura 3.3: Implementazione MATLAB DFT.

3.3 DFT di segnali analogici

A partire da un segnale nel tempo $x(t)$ è possibile simularne una versione campionata nell'intervallo frequenza di campionamento:

$$f_c = \frac{1}{T_0} \tag{3.3}$$

nel seguente modo:

$$x[n] = x(nT_c) \tag{3.4}$$

dove $N = T_0 f_c$ è il numero di campioni. Si considerano, dunque, i seguenti segnali:

- $x(t) = sinc^2(t)$
- $x(t) = e^{-4|t|}$
- $x(t) = cos(2\pi t)$

Capitolo 4

LAB3: Periodogrammi