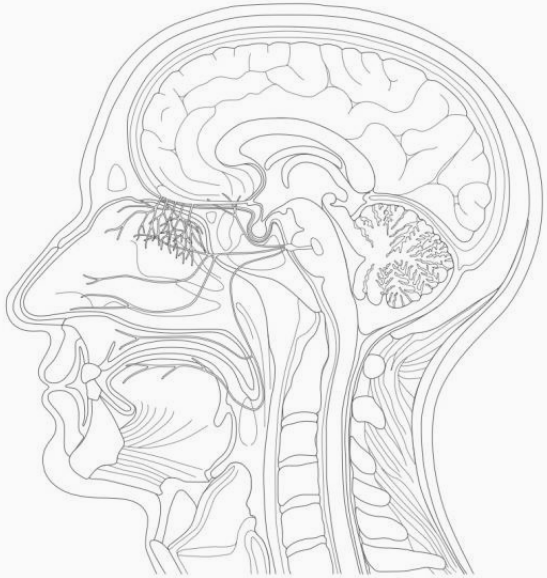
Introduzione a Matlab



Tutor: Dr. Giovanna Nordio e Giulia Vallini

Prof. Mattia Veronese
Email: mattia.veronese@unipd.it
Dipartimento di Ingegneria dell'Informazione

Ricevimento: su appuntamento (e-mail)
Edificio DEI/A, piano 2o, stanza 214

A cosa serve MATLAB?

MATLAB e' un linguaggio di calcolo matriciale e vettoriale

Funzionalità MATLAB

Analisi dei dati

Esplora, modella e visualizza dati

Grafica

Visualizzazione ed esplorazione dei dati

Sviluppo di algoritmi

Progetta algoritmi per applicazioni desktop ed embedded

Creazione di app

Creazione di app web e desktop

Utilizzo di MATLAB con altri linguaggi

Utilizzo di MATLAB con Python, C/C++, Fortran, Java e altri linguaggi

Hardware

Connetti MATLAB all'hardware

Calcolo parallelo

Esegui calcoli su larga scala utilizzando computer desktop multicore, GPU, cluster, grid e cloud

Distribuzione su desktop e via web

Condividi i tuoi programmi MATLAB

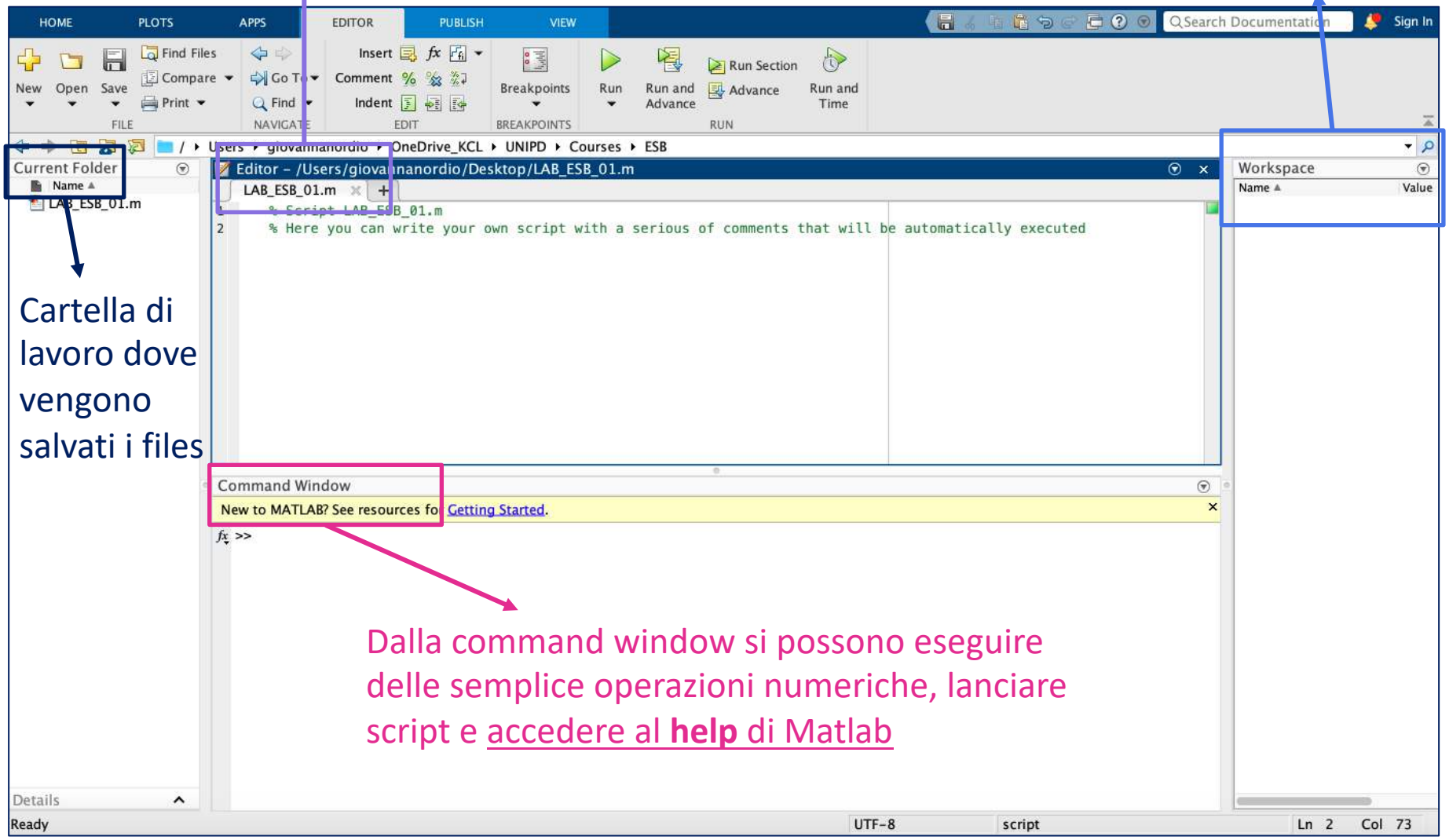
Cloud computing

Lavora in ambienti cloud da MathWorks Cloud a cloud pubblici, inclusi AWS e Azure

L'interfaccia grafica di Matlab

Spazio di lavoro dove vengono memorizzate le variabili create o importate in Matlab da file/programmi esterni

Matlab script



Cartella di lavoro dove vengono salvati i files

Dalla command window si possono eseguire delle semplice operazioni numeriche, lanciare script e accedere al **help** di Matlab

Matlab tutorials

Per chi non ha esperienza con Matlab:

https://matlabacademy.mathworks.com/

MATLAB Onramp
MATLAB Fundamentals

N.B. per accedere dovete creare un account Mathworks (con la vostra mail unipd)

Altri tutorials utili:

- https://people.ucalgary.ca/~ranga/enel563/matlab_tutorial.pdf (STESSO DEL LIBRO Biomedical Signal Analysis)
- https://pselab.chem.polimi.it/wp-content/uploads/2017/03/SECDIC-Tutorial-Matlab2.pdf

Script e funzione

SCRIPT

Uno script è un file .m che contiene piú linee di comando e operazioni, eseguibili automaticamente tramite chiamata diretta.

FUNZIONE

Una funzione è un programma che, avendo a disposizione una serie di dati di input, fornisce una serie di dati di output, effettuando delle operazioni. Una funzione esterna permette di salvare una serie di operazioni che vengono ripetute frequentemente, richiamandola negli script SOLO quando necessario (es. EoS, sistemi di equazioni algebriche o differenziali,...).

```
function [OUTPUT_1,…,OUTPUT_N] = myfunction(INPUT_1,…,INPUT_M)
```

Alcuni comandi utili

Consiglio - Iniziare uno script con:

clear all: pulisce il Workspace, cancellando tutte le variabili salvate
close all: chiude i grafici aperti
clc: pulisce i comandi scritti nella Command Window

Alcune strategie per scrivere uno script/funzione ordinata ed efficiente:

- Commenta il piu' possibile il tuo codice
- Raggruppa linee di codice in base alla loro funzionalita'
- Fai il backup del tuo codice regolarmente
- Usa nomi di variabili che hanno senso

NB Matlab dispone di una ricca libreria di funzioni, a cui si può accedere tramite **help menu** (help *#nomefunzione*), oppure tramite documentazione online.

Calcolo matriciale e vettoriale

MATLAB è un linguaggio di calcolo matriciale e vettoriale

Queste sono solo alcune delle operazioni utili da ricordare:

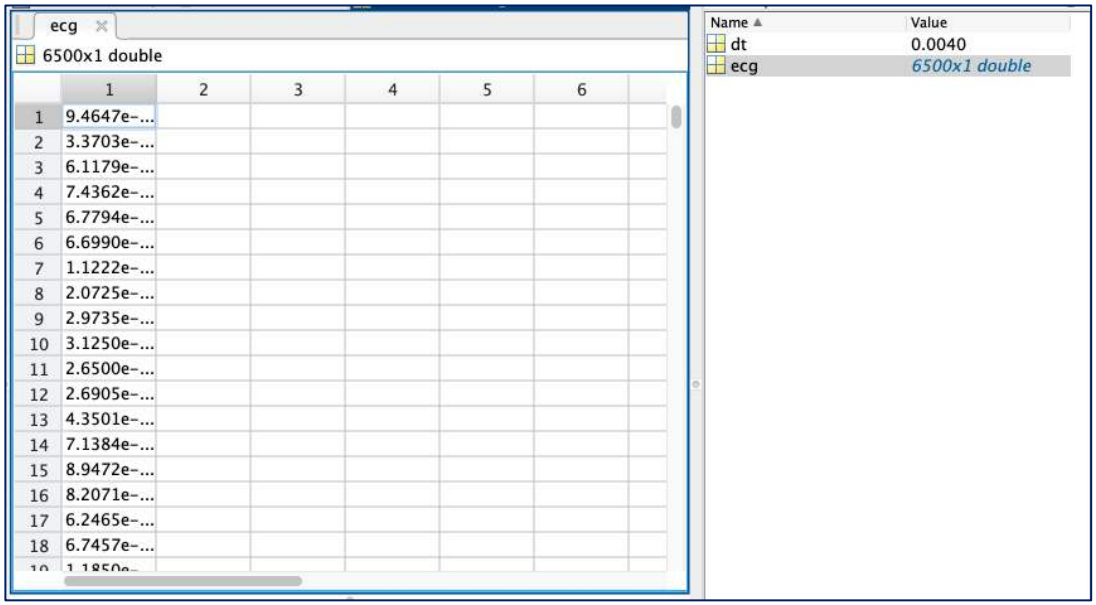
OPERATION	MATLAB COMMAND
Matrice 2x3	>> A = [1 5 7; 2 9 8] 1 5 7 2 9 8
Matrice vuota 2x3	>> A = zeros(2,3) 0 0 0 0 0 0
Vettore colonna vuoto 3x1	>> v = zeros(3,1) 0 0 0
Vettore riga di solo uno 1x3	>> v = ones(1,3) 1 1 1
Vettore di punti equispaziati	>> v = 1:5 1 2 3 4 5 >> v = -1.1:0.4:0.2 -1.1000 -0.7000 -0.3000 0.1000

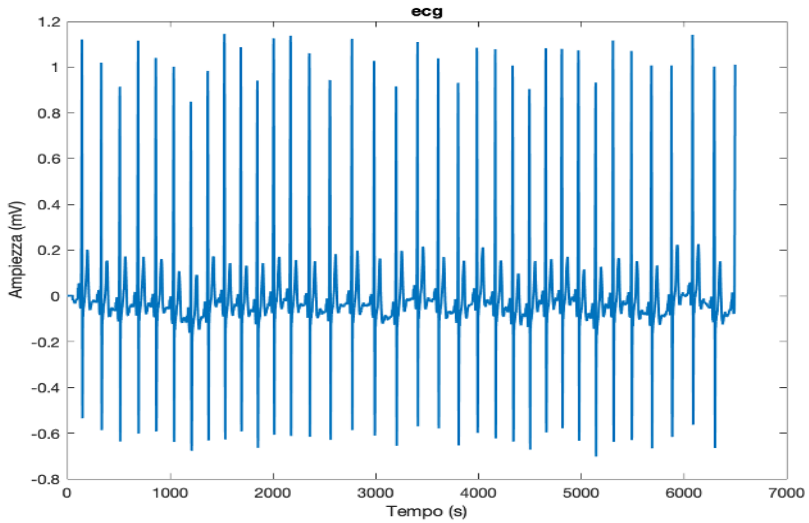
Calcolo matriciale e vettoriale

Un segnale biomedico campionato è un classico esempio di array (vettore).

Caso ECG

ECG misurato è un vettore lungo 6500 campioni





Rappresentazione grafica di un segnale

Matlab offre una ricca libreria per rappresentare graficamente un segnale.
Per un segnale bidimensionale, in analogia ad un segnale di una variabile dove l'ordinata è un vettore, che rappresenta la funzione valutata per N punti dell'ascissa, l'ordinata è una matrice, che rappresenta la funzione valutata per NxM punti delle due variabili indipendenti x e y.

```
plot(x, y, 'objects', value)
```

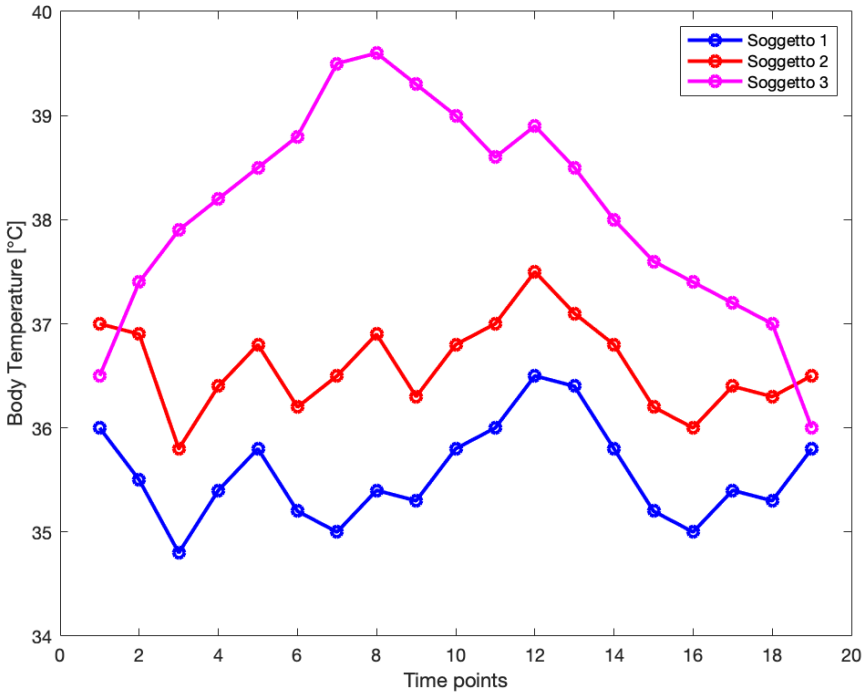
Proprietà:

- linewidth
- markersize
- markeredgecolor
- markerfacecolor

Valore: dipende dalla proprietà

- Dimensione in punti
- Colori

```
% Plot
figure(1)
plot(thermo(:,1), '-bo', 'LineWidth',2)
hold on
plot(thermo(:,2), '-ro', 'LineWidth',2)
hold on
plot(thermo(:,3), '-mo', 'LineWidth',2)
xlabel('Time points')
ylabel('Body Temperature [°C]')
legend('Soggetto 1','Soggetto 2', 'Soggetto 3')
```

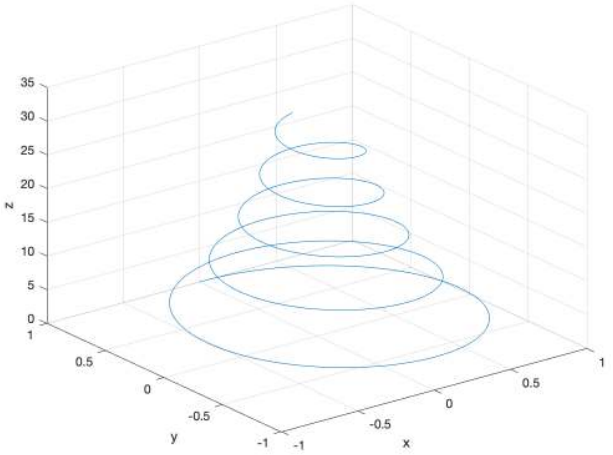


Rappresentazione grafica di un segnale

Per segnali tridimensionali:

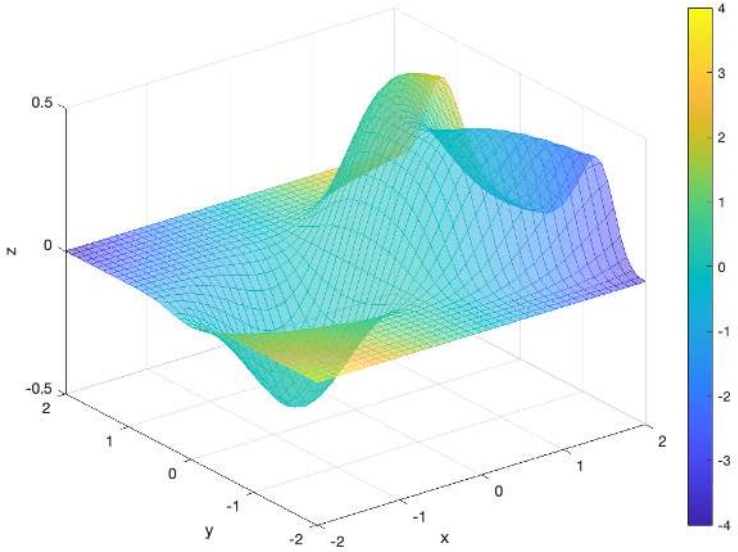
- `plot3` → per tracciare una linea in uno spazio tridimensionale

```
figure(2)
t = [0:pi/50:10*pi];
plot3(exp(-0.05*t).*sin(t),exp(-0.05*t).*cos(t),t)
xlabel('x')
ylabel('y')
zlabel('z')
grid
```



- `mesh` → per tracciare una superficie 3D

```
figure(3)
[X,Y] = meshgrid(-2:0.1:2);
Z = X.*exp(-((X-Y.^2).^2+Y.^2));
C = X.*Y;
s = mesh(X,Y,Z,C,'FaceAlpha','0.5')
s.FaceColor = 'flat';
xlabel('x')
ylabel('y')
zlabel('z')
colorbar
```



Esercizio 1

Creare il vettore riga v = (5, 9, -6, 0, 0.2, 3.6) e il vettore colonna w = (0, 0.1, -5, -3, 2.3, 9)

1. Salvare in A la somma di v e w
2. Salvare in B il prodotto scalare di v per w (usare il commando **dot**)
3. Salvare in C gli elementi di posto pari di v
4. Salvare in D cinque copie del vettore v (suggerimento: usare il comando **kron**)
5. Salvare in E gli elementi di posto multiplo di 4 di D
6. Salvare in F la matrice w*v
7. Salvare in G la matrice costituita dalla seconda fino alla quarta riga di F e dalla prima fino alla terza Colonna di F

Esercizio 2

Il file DATA_Lab03_Es02.xlsx contiene una tabella di 3 colonne e 130 righe, dove la prima colonna corrisponde alla temperatura corporea (in Fahrenheit) di 130 soggetti, la seconda colonna è il sesso dei soggetti (0=uomo, 1=donna), e la terza colonna e' il battito cardiaco dei 130 soggetti.

1. Carica il file contenente i dati

   N.B. Per importare un file in Matlab è possibile usare una di queste alternative

   - **load filename.extension**
   - **xlsread('filename')** per file excel
   - **importdata('filename')**

2. Crea due matrici (130x2), una per gli uomini e una per le donne, con la corrispondente temperatura e battito cardiaco (suggerimento: usa il comando **find**)
3. Rappresenta graficamente la temperatura corporea in funzione del battito cardiaco per gli uomini e per le donne in due grafici separati (usa il comando **subplot**)

Esercizio 3

1. Scrivere un M-file che disegni con una linea continua rossa la funzione **$f(t)=2e^{(-0.05t)}$** nell'intervallo [0,100] (rappresentare f(t) con 500 punti equispaziati; usare il comando **linspace**)

2. Rappresentare f(t) nel riquadro superiore di una figura a due riquadri (suggerimento: usare **subplot**). Nel riquadro inferiore rappresentare di nuovo la stessa funzione con sovrapposti, rappresentati da cerchietti blu, dei suoi campioni rumorosi generati artificialmente ai tempi (0, 5, 10, 15, …100). Per generare i campioni considerare rumore gaussiano additivo a media nulla e standard deviation 0.03 (suggerimento: usare funzione **randn**). Intitolare i grafici rispettivamente con *'Esponenziale decrescente'* ed *'Esponenziale decrescente con campioni rumorosi (SD=0.03)'.*

3. Acquisire da tastiera (suggerimento: usare l'istruzione input) un valore positivo (suggerimento: usare una struttura **while … end** per controllare che l'utente immetta un valore SD strettamente positivo) da usare come standard deviation del rumore additivo da usare nella generazione dei campioni rumorosi di f(t). Intitolare i grafici dei due riquadri rispettivamente con "Esponenziale decrescente" e "Esponenziale decrescente con campioni rumorosi (SD=__)" (far apparire nel titolo del secondo riquadro il valore della SD immesso da tastiera, suggerimento: usare **num2str**)

Esercizio 4 – da fare a casa

Data una matrice A, le cui dimensioni sono *m* per le righe ed *n* per le colonne, creare una matrice B contenente gli stessi elementi della matrice A, ma con le righe pari ordinate in modo crescente e le righe dispari ordinate in maniera decrescente.

Suggerimenti:

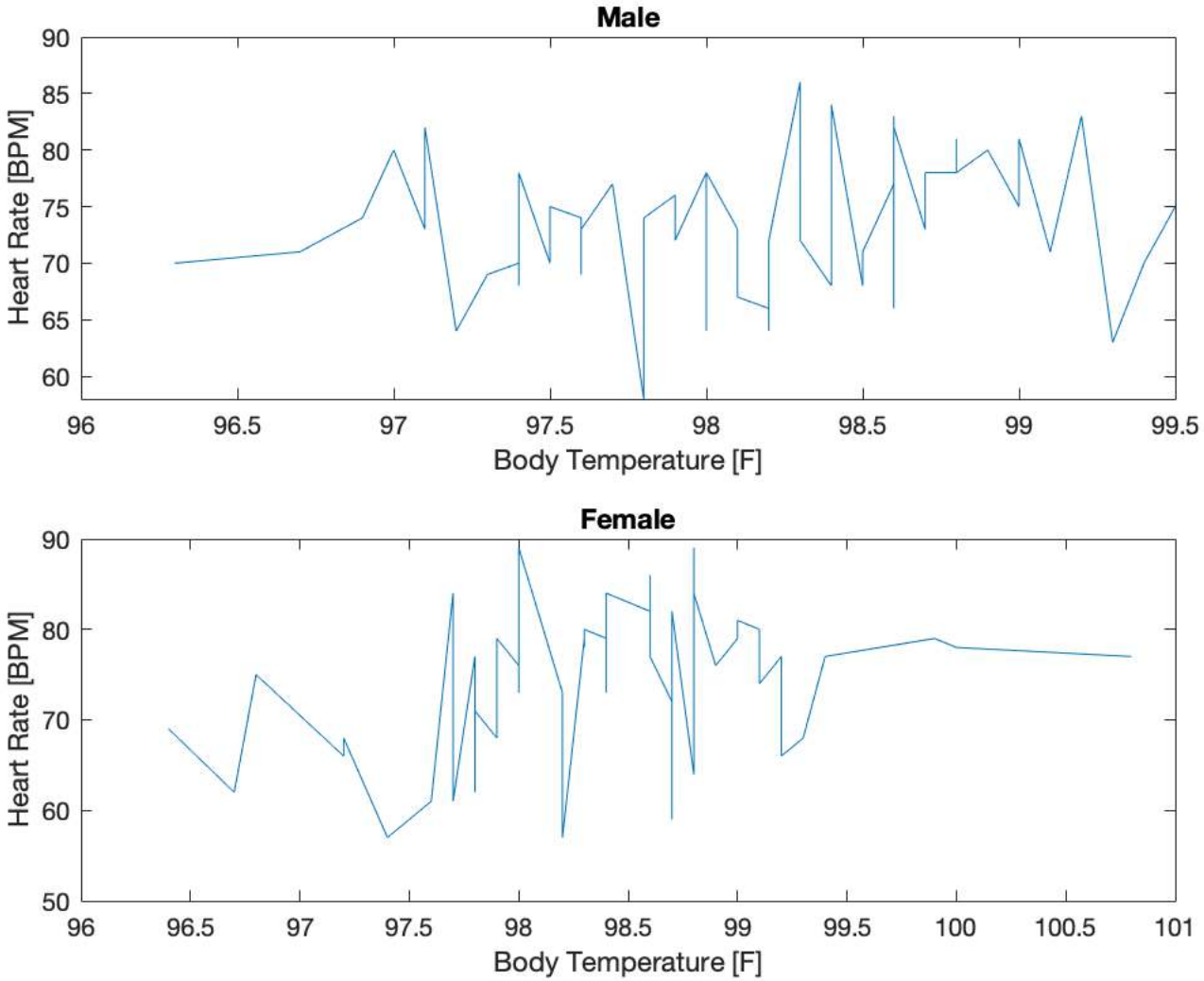
- crea una funzione ordina_crescente
- crea una funzione ordina_decrescente

SOLUZIONI

Soluzioni - Esercizio 1

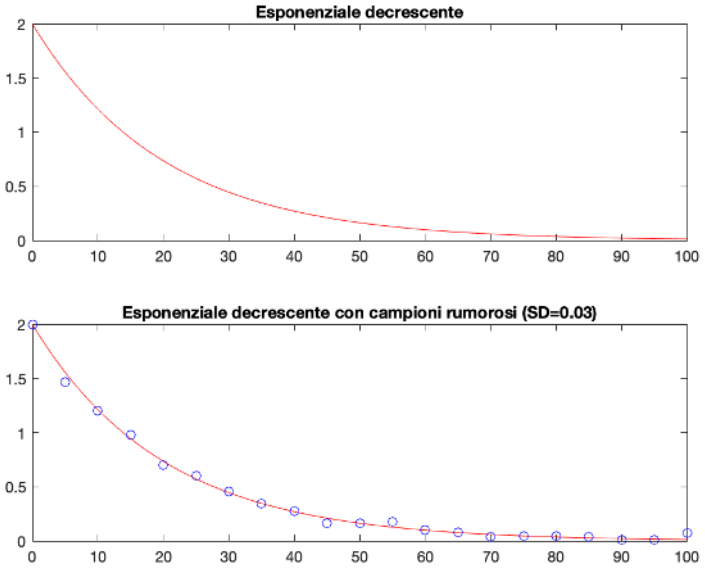
```
% Vettore riga v
v = [5, 9, -6, 0, 0.2, 3.6]; % Altrimenti v = [5 9 -6 0 -0.2 3.6];

% Vettore colonna W
w = [0, 0.1, -5, -3, 2.3, 9]'; % Altrimenti v = [0; 0.1; -5; -3; 2.3; 9];

% Salvare in A la somma del vettore riga v e  del vettore colonna w
A = v' + w;

% Salvare in B il prodotto scalare di v per w (usare il commando dot)
B = dot(v',w);

%Salvare in C gli elementi di posto pari di v
C = v(2:2:end);

%Salvare in D cinque copie del vettore v
D = kron(ones(5,1),v);

%Salvare in E gli elementi di posto multiplo di 4 di D
E = D(4:4:end);

%Salvare in F la matrice w*v
F = w*v;

%Salvare in G la matrice costituita dalla seconda fino alla quarta riga di
%F e dalla prima fino alla terza Colonna di F
G = F(2:4,1:3);
```

Soluzioni – Esercizio 2



Soluzioni - Esercizio 3

```
t = linspace(0,100,500)';
f = 2*exp(-0.05*t);
subplot(2,1,1)
plot(t, f, 'r')
title('Esponenziale decrescente')

ts = [0:5:100]';
fs = 2*exp(-0.05*ts);
vs = 0.03*randn(length(fs),1);
ys = fs + vs;

subplot(2,1,2)
plot(t, f, 'r')
hold on
plot(ts, ys, 'ob')
title('Esponenziale decrescente con campioni rumorosi (SD=0.03)')
```




```
sd = input('Inserisci un valore di SD: ');
while sd <= 0

    sd = input('Dammi un valore positivo di SD: ');

end % while

figure(2)
subplot(2,1,1)
plot(t, f, 'r')
title('Esponenziale decrescente')

ts = [0:5:100]';
fs = 2*exp(-0.05*ts);
vs = sd*randn(length(fs),1);
ys = fs + vs;

subplot(2,1,2)
plot(t, f, 'r',ts, ys, 'ob')
title(['Esponenziale decrescente con campioni rumorosi (SD='+convertCharsToStrings(num2str(sd))+')'])
```

Soluzioni - Esercizio 4

```
function [ M ] = ordina_crescente( M, r, c )
%ordina_crescente funzione che implementa l'algoritmo del selection sort
%   La funzione accetta 3 parametri (la matrice, l'indice della riga da
%   ordinare e il numero di colonne) e restituisce la matrice con la
%   r-esima riga ordinata in modo crescente

    for i = 1 : c-1
        i_min = i;
        j = i+1;
        for j = j : c
            if M(r, j) < M(r, i_min)
                % Se vero, aggiorno l'indice dell'elemento minimo
                i_min = j;
             end
         end

        % Blocco di codice relativo allo scambio
        temp = M(r, i);
        M(r, i) = M(r, i_min);
        M(r, i_min) = temp;
    end
end
```

```
function [ M ] = ordina_decrescente( M, r, c )
 %ordina_decrescente funzione che implementa l'algoritmo del selection sort
%   La funzione accetta 3 parametri (la matrice, l'indice della riga da
%   ordinare e il numero di colonne) e restituisce la matrice con la
%   r-esima riga ordinata in modo decrescente

    for i = 1 : c-1
        i_max = i;
        j = i+1;
        for j = j : c
            if M(r, j) > M(r, i_max)
                i_max = j;
             end
        end

        temp = M(r, i);
        M(r, i) = M(r, i_max);
        M(r, i_max) = temp;
    end
end
```

Soluzioni - Esercizio 4

```
%% Esercizio 4
% Input la dimensione righe della matrice A
flag=true;
while flag==true
    m = input('Inserisci numero di righe per la matrice A:');
    flag = m<=0;
end

% Input la dimensione colonne della matrice A
flag=true;
while flag==true
    n = input('Inserisci numero di colonne per la matrice A:');
    flag = n<=0;
end

% Input gli elementi della matrice A
disp('Inserisci i valori della matrice A (scrivi un valore e premi invio)')
A=[];
for i=1:m
    for j=1:n
        A(i,j)=input('');
    end
end

% Creo la matrice B esattamente uguale alla matrice A
B = A;

for i = 1 : m
    if mod(i,2)==0
        % Caso riga pari
        B = ordina_crescente(B, i, n);
    else
        % Caso riga dispari
        B = ordina_decrescente(B, i, n);
    end
end

disp(A)
disp(B);
```