MANUALETTO PER MATLAB

PREMESSA

Questo manualetto per ***MATLAB*** nasce e si sviluppa contemporaneamente all'attività svolta dagli autori per inserire in modo graduale nel curricolo del Corso di Sistemi Elettronici Automatici lo studio e l'impiego di questo software che oggi viene utilizzato diffusamente in ambiente sia universitario che industriale per il calcolo scientifico, per l'analisi e la rappresentazione dei dati, per l'analisi e la progettazione di apparati anche complessi. Nelle intenzioni degli autori, questo lavoro servirà solo a mettere in evidenza e a memorizzare alcune funzioni e procedure di MATLAB che torneranno molto utili, nel corso dell'intero triennio, per rendere più semplici ed "amichevoli" per gli allievi i concetti da affrontare nel corso di Sistemi. Sarà quindi un'opera senza tante pretese e da aggiornare in continuazione col progredire degli studi.

GENERALITÀ

All'apertura, MATLAB presenta una serie di finestre:

a) ***Workspace*** (ancora da interpretare):
b) ***Current directory***: è la cartella corrente in cui salvare i file di lavoro o da cui prelevarli
c) ***Command history***: riporta tutta la sequenza di comandi dati a MATLAB fino al momento attuale
d) ***Command window (CW)***: è la finestra principale; quella utilizzata per le operazioni. In tale finestra possiamo scrivere un comando e farlo eseguire all'istante da MATLAB. Possiamo, in alternativa, scrivere a parte un file di testo, con estensione ***.m*** (***M-file***), in cui raccogliere una intera sequenza di comandi e chiedere a MATLAB di eseguire l'intero programma così scritto. ***Attenzione: il nome degli M-file non deve contenere spazi.***

MANUALETTO

Interprete di comandi

La ***Command window*** *agisce come un interprete di comandi, nel senso che ogni comando in essa scritto viene subito eseguito premendo il tasto* ***INVIO***.

- **clc** : ***pulisce la Command window***.
- **clf** : ***pulisce la finestra dei grafici***.
- **;** : il ***punto e virgola*** dopo un comando ***evita la pubblicazione dei risultati sulla Command window.***

Vettori e Matrici

- ***Array:*** è un insieme di dati omogenei, cioè dello stesso tipo
  - ***Vettore***: Array ad una dimensione. Può essere costituito o da una unica riga di dati (***vettore riga***) o da una unica colonna di dati (***vettore colonna***). Salvo indicazione contraria, quando parleremo di vettore ci riferiremo ad un vettore riga.
  - ***Matrice***: Array a due dimensioni. Una matrice di dimensioni "***n*m***" ha ***n*** righe ed ***m*** colonne.
- **r = [1 4 2 6]** : costruisce un ***vettore riga***, indentificato con la variabile ***r***, con i 4 valori "***1,4,2,6***" indicati tra parentesi quadre.
- **r = [5 ; 2 ; 0 ; 1]** : costruisce un ***vettore colonna***, indentificato con la variabile ***r***, con i 4 valori "***5,2,0,1***" indicati tra parentesi quadre. I punti e virgola tra i valori delimitano le righe, che in questo caso sono costituite da un solo valore.

- **t = [v_i : p : v_f]** : costruisce un vettore, identificato con ***t***, formato da numeri reali che partono da v_i e raggiungono il valore finale v_f con un passo pari a ***p***. La variabile ***t*** spesso viene usata per indicare istanti di tempo.
  Es.: **t = [0 : 0.01 : 10]** ➔ Il vettore ***t*** è costituito da istanti (numeri reali) che partono da ***0*** e raggiungono il valore finale ***10*** con passo ***0,01*** ➔ t = {0 - 0,01 – 0,02 – 0,03 - …… - 9,98 – 9,99 – 10}.
- **r = linspace(v_i , v_f , n)** : costruisce il vettore ***r*** con ***n*** valori, equamente distanziati tra loro, a partire da v_i e fino a v_f .
  Es.: **r = linspace(0, 1, 3)** ➔ r = {0 - 0,5 – 1}.
- **M = [1 2 3 ; 4 5 6 ; 7 8 9]** : costruisce la matrice ***M*** le cui ***righe*** sono indicate in parentesi quadre, in sequenza e separate da un ***punto e virgola***.

$$M = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{matrix}$$

- **D = det(M)** : calcola il ***determinante*** della matrice ***M***.
- **A = inv(M)** : calcola la ***matrice inversa*** della matrice ***M***.

Funzioni

- **y = sin(ω * t)** : consente di costruire il vettore ***y*** di valori della funzione ***seno***. Lo scalare ***ω*** rappresenta la pulsazione; il vettore ***t*** contiene tutti i valori della variabile tempo.
- **y = cos(ω * t)** : consente di costruire il vettore ***y*** di valori della funzione ***coseno***. Lo scalare ***ω*** rappresenta la pulsazione; ***t*** è un vettore e contiene tutti i valori della variabile tempo.
- **y = 3*t** : consente di costruire il vettore ***y*** di valori della funzione ***y = 3t***. Il vettore ***t*** contiene tutti i valori della variabile tempo.
- **y = t.^2** : il risultato di questa operazione è il vettore ***y*** i cui elementi si ottengono elevando al quadrato ogni elemento del vettore ***t***.
  Il ***punto*** subito prima di un operatore aritmetico ***distingue le operazioni aritmetiche sull'array*** (eseguite cioè elemento per elemento) ***da quelle sui vettori o sulle matrici*** (eseguite come previsto nell'***algebra lineare)***.
  Es. **a = b.*b** ha significato perché ogni elemento di b viene moltiplicato per se stesso.
  a = b*b non ha significato perché si chiede di moltiplicare, riga per colonna, due vettori dello stesso tipo (riga o colonna).
  Il punto può essere trascurato quando non sussistono ambiguità.
  Es. ***c = a+b = a.+b*** in questo caso il punto può essere trascurato perché entrambe le operazioni danno lo stesso risultato: un vettore ***c*** i cui elementi si ottengono ***sommando gli elementi omologhi di a e b***.

Polinomi

- **p = [1 5 4 10]** : un polinomio viene rappresentato con un vettore che contiene i suoi coefficienti presi nell'ordine decrescente delle potenze della variabile indipendente. Nell'esempio dato si ha $p = x^3 + 5x^2 + 4x + 10$. Se nel polinomio manca qualche termine, occorre tener conto di questo fatto mettendo degli 0 al posto dei coefficienti mancanti. Es. **n = [2 0 3 0 6]** rappresenta il polinomio $n = 2x^4 + 3x^2 + 6$
- **p = conv(p_1,p_2)** : il polinomio ***p*** viene ottenuto dal ***prodotto*** dei polinomi p_1 ***e*** p_2 . L'operazione "conv" agisce solo tra due polinomi; la stessa operazione su più di due polinomi va spezzettata in operazioni parziali. Es. **p = conv(conv(p_1,p_2),p_3)** esegue il ***prodotto tra*** p_1 ***e*** p_2 ***e poi moltiplica il risultato per*** p_3.

- **[q,r] = deconv(v,u)** : esegue il ***rapporto tra i polinomi v ed u***; il ***quoziente*** è salvato nel vettore ***q*** mentre il ***resto*** è salvato nel vettore *r*.
- **r = roots(p)** : calcola in ***r*** le ***radici del polinomio p***.
- **p = poly(r)** : calcola in ***p*** i ***coefficienti del polinomio*** che ha per ***radici*** i valori contenuti nel vettore ***r*** .
- **[r , p, k] = residue (n , d)** : calcola ***residui (r), poli (p) e i coefficienti del termine diretto (k)*** di una funzione razionale fratta, i cui coefficienti del numeratore e del denominatore siano raccolti, rispettivamente, in ***n*** e ***d***. I residui sono i coefficienti dello sviluppo in frazioni parziali della funzione razionale fratta; i poli sono le radici del denominatore; il termine diretto è il risultato della divisione tra il numeratore ed il denominatore, quando il primo ha un grado maggiore o uguale a quello del secondo. Se la funzione razionale fratta è data sotto forma fattorializzata, occorre portarla nella forma di rapporto di polinomi utilizzando la funzione ***conv***.
- **[n , d] = residue (r , p, k)** : calcola i coefficienti del numeratore e del denominatore (***n*** e ***d***) di una funzione razionale fratta a partire da ***residui (r), poli (p) e coefficienti del termine diretto (k).***

Funzione di trasferimento

- **F = tf(n , d)** : consente di creare un funzione di trasferimento, di eseguire rapidamente prodotti e divisioni di funzioni, di calcolare la funzione di trasferimento di blocchi in cascata o in retroazione.
  Es. **G = tf(10, [3 4]) ; H = tf(0.1,[1 100]) ; W = G/1+(H*G)**

$$G(s) = \frac{10}{3s+4} \quad ; \quad H(s) = \frac{0,1}{s+100} \quad ; \quad W = \frac{3,333s + 333,7}{s^2 + 101,3s + 133,3}$$

- **F = minreal(F)** : riporta una funzione razionale fratta nella sua ***forma minima***. In altre parole, ***cancella zeri e poli uguali***.
- **W = feedback(G,H,±1)** : calcola la funzione di trasferimento ***a ciclo chiuso (W)*** a partire da quelle, rispettivamente, della ***linea diretta (G)*** e della ***linea di retroazione (H)***. L'ultimo parametro (**±1)** indica ***retroazione positiva*** quando vale ***+1***; ***retroazione negativa*** quando vale ***-1***. In quest'ultimo caso (***retroazione negativa***) tale parametro può essere omesso.
  Es. **W = feedback(G,H)**
- **C = series(A,B)** : calcola la funzione equivalente ***C*** dei due blocchi in ***cascata A*** e ***B***.
- **C = parallel(A,B)** : calcola la funzione equivalente *C* dei due blocchi in ***parallelo A*** e ***B***. Vale solo per la somma; per la differenza occorre far ricorso a qualche artifizio.
- **s = tf('s')** : dichiarazione che fa in modo che tutte le funzioni di trasferimento utilizzate successivamente siano nella forma ***rapporto di polinomi***.
- **F = zpk(z, p, k)** : costruisce la funzione di trasferimento F(s) ***a partire da zeri, poli e guadagno***, che sono raccolti, rispettivamente, nei ***vettori z, p*** e nello ***scalare k***. F(s) viene quindi rappresentata in una ***forma fattorializzata***. Attenzione: il guadagno ***k non è il guadagno statico***, ma la costante che compare nella F(s) quando questa è espressa nella forma fattorializzata normale. Se si parte quindi da una forma di Bode, questa deve essere convertita nella forma normale. Quando, infine, nella F(s) non sono presenti zeri, essi vanno indicati con ***l'insieme vuoto* [ ]**; lo stesso dicasi per i poli.

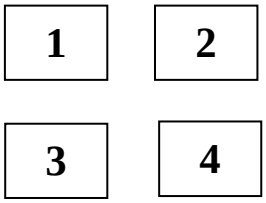
$$\textbf{Es.:} \quad F(s) = \frac{10}{(1+s0,1)(1+s0,01)} = \frac{10}{0,1(\frac{1}{0,1}+s)0,01(\frac{1}{0,01}+s)} = \frac{10}{0,001(10+s)(100+s)} =$$

$$= \frac{10000}{(s+10)(s+100)} \quad \Rightarrow \quad z = [\,] \quad ; \quad p = [-10 \quad -100] \quad ; \quad k = 10000$$

- **s = zpk('s')** : dichiarazione che fa in modo che tutte le funzioni di trasferimento utilizzate successivamente siano nella forma ***contenenti zeri, poli, guadagno***.
- **f = ilaplace(F, s, t)** : calcola ***l'antitrasformata f(t) della funzione F(s)***. E' necessario che prima del comando venga riportata la dichiarazione ***Syms s t*** La funzione F(s) ***non*** deve essere generata con il metodo ***tf*** ; può essere invece rappresentata nella forma ***fattorializzata***.

Grafici

- **subplot(r c p)** : costruisce una matrice di grafici con dimensioni massime ***2 * 2***. Lo scalare ***r*** rappresenta il numero di ***righe***; lo scalare ***c*** rappresenta il numero di ***colonne***; lo scalare ***p*** rappresenta la ***posizione*** del grafico nell'ambito della matrice, secondo una numerazione crescente che va da sinistra a destra e dall'alto in basso.



- **plot(x,y)** : disegna il grafico di ***y = f(x)***.
- **xlabel('***testo***')** : definisce la grandezza rappresentata sull'asse ***x*** dell'ultimo grafico tracciato.
- **ylabel('***testo***')** : definisce la grandezza rappresentata sull'asse ***y*** dell'ultimo grafico tracciato.
- **gtext('***testo***')** : consente di inserire una ***scritta*** su un grafico con l'aiuto del ***mouse***.
- **title('***testo***')** : scrive il ***titolo*** in testa all'ultimo grafico tracciato.
- **zoom on** : consente di ***ingrandire*** (con il tasto ***sinistro*** del mouse) o di ***rimpicciolire*** (con il tasto ***destro*** del mouse) la zona di un grafico posta in corrispondenza della ***posizione del cursore*** del mouse.
- **grid** : Aggiunge la ***griglia*** all'ultimo grafico tracciato.
- **semilogx(x,y)** : rappresenta la funzione ***y = f(x)*** su un grafico che ha l'asse ***x in scala logaritmica in base 10*** e l'asse ***y in scala lineare***.
- **semilogy(x,y)** : rappresenta la funzione ***y = f(x)*** su un grafico che ha l'asse ***x in scala lineare*** e l'asse ***y in scala logaritmica in base 10***.
- **loglogy(x,y)** : rappresenta la funzione ***y = f(x)*** su un grafico che ha ***sia l'asse x che l'asse y in scala logaritmica in base 10***.
- **ltiview** : consente di osservare, in base alla funzione di trasferimento del sistema considerato,
    a) ***Risposta al gradino unitario***
    b) ***Risposta all'impulso***
    c) ***Diagrammi di Bode***
    d) ***Diagrammi di Nyquist***
    e) ***Carta di Nichols***
    f) ***Posizione dei poli e degli zeri***

  Per ogni tipo di grafico è possibile visualizzare il valore di determinati parametri.
  All'apertura della finestra occorre importare la funzione di trasferimento da visualizzare (per importare più di una funzione usare il tasto Control); successivamente col tasto destro del mouse si possono effettuare tutta una serie di scelte. Per la rappresentazione contemporanea di più funzioni, si può, in alternativa a quanto detto prima, indicare il nome delle funzioni da rappresentare come una lista di parametri all'atto della chiamata della funzione ***ltiview***.
  Es. ***ltiview(F, G, H)*** : consente di visualizzare i grafici delle tre funzioni F, G, H.
  Nota: le funzioni da rappresentare devono essere create con il metodo ***tf***.
- **bode** : consente di tracciare i diagrammi di Bode. Ci sono vari modi di richiamare questa funzione grafica; a noi interesseranno solo alcuni di questi.

a) ***bode(F)*** : traccia il diagramma di Bode della funzione F(s):
b) ***bode(G, H, W)*** : traccia i diagrammi sovrapposti delle tre funzioni G(s), H(s) e W(s). I diagrammi sono tracciati con colori diversi: per individuare la corrispondenza tra colore e funzione aprire il menù contestuale e soffermarsi sulla voce "Systems".
c) ***bode(G , 'stile1' , H , 'stile2' , W , 'stile3')*** : traccia i diagrammi sovrapposti delle tre funzioni G(s), H(s) e W(s). Ognuna delle stringhe 'stile1', 'stile2', e 'stile3' consente di impostare, per la funzione che le precede, il colore, il tipo di linea e la presenza di markers. Es. ***bode(G , 'r' , H , 'y --' ,F , 'b - .' , W , 'g .')*** : traccia
   - per ***G*** una curva ***continua*** di colore ***rosso***
   - per ***H*** una curva ***tratteggiata*** di colore ***giallo***
   - per ***F*** una curva ***tratto e punto*** di colore ***blue***
   - per ***W*** una curva costituita solo da ***markers*** (**.**) di colore ***verde***
d) ***[m, f] = bode (F, w)*** : Calcola ***modulo*** e ***fase*** della funzione ***F*** in corrispondenza della pulsazione ***w***. Il modulo è ottenuto in assoluto; per trasformarlo in ***dB***, si può usare la funzione Log10 → ***a = 20*log10(a)***.

*Nota: le funzioni da rappresentare devono essere create con il metodo **tf**.*

- **margin** : calcola i margini di guadagno e di fase di una funzione di trasferimento a ciclo aperto. Di seguito sono indicati i due modi che noi useremo per richiamare tale funzione.
  a) ***margin(F)*** : traccia automaticamente i diagrammi di ***Bode*** della funzione e su di essi indica i ***margini di guadagno (G_m) e di fase (P_m)***. Il margine di guadagno G_m è espresso in *dB*.
  b) ***[G_m , P_m , w_g , w_p] = margin(F)*** : determina margine di guadagno (G_m) e margine di fase ($\mathbf{P_m}$) con le corrispondenti pulsazioni di attraversamento w_g e w_p . Il margine di guadagno G_m è espresso in ***assoluto***.
- **nyquist** : consente di tracciare i diagrammi di Nyquist. Ci sono vari modi di richiamare questa funzione grafica; a noi interesseranno solo alcuni di questi.
  a) ***nyquist(F)*** : traccia il diagramma di Nyquist della funzione F(s):
  b) ***nyquist(G, H, W)*** : traccia i diagrammi sovrapposti delle tre funzioni G(s), H(s) e W(s). I diagrammi sono tracciati con colori diversi: per individuare la corrispondenza tra colore e funzione aprire il menù contestuale e soffermarsi sulla voce "Systems".
  c) ***nyquist(G , 'stile1' , H , 'stile2' , W , 'stile3')*** : traccia i diagrammi sovrapposti delle tre funzioni G(s), H(s) e W(s). Ognuna delle stringhe 'stile1', 'stile2', e 'stile3' consente di impostare, per la funzione che le precede, il colore, il tipo di linea e la presenza di markers. Es. ***nyquist(G , 'r' , H , 'y --' ,F , 'b - .' , W , 'g .')***: traccia
     - per ***G*** una curva ***continua*** di colore ***rosso***
     - per ***H*** una curva ***tratteggiata*** di colore ***giallo***
     - per ***F*** una curva ***tratto e punto*** di colore ***blue***
     - per ***W*** una curva costituita solo da ***markers*** (**.**) di colore ***verde***
  d) ***[r, i] = nyquist (F, w)*** : Calcola ***parte reale*** e ***parte immaginaria*** della funzione F in corrispondenza della pulsazione ***w***.

  *Nota: le funzioni da rappresentare devono essere create con il metodo **tf**.*
- **figure** : apre una nuova finestra su cui disegnare un nuovo grafico.
  figure(n) : consente di aprire e mettere in evidenza la finestra relativa al grafico con numero d'ordine ***n***. Se il grafico non esiste, crea una finestra vuota associando ad essa il numero d'ordine ***n***.

M-File

Nota: gli esempi proposti nel seguito possono essere "copiati ed incollati" direttamente nell'ambiente di lavoro Matlab per creare un M-file funzionante. Occorre solo stare attenti che qualche pezzo di commento non perda il simbolo "%" in prima posizione che lo distingue dai comandi ordinari.

- **% TRACCIA IL GRAFICO DELLE FUNZIONI Y1=3T, Y2=T^2, Y3=SEN(WT), Y4=COS(WT)**

```
%Genera il vettore t dei tempi
t=[0:0.01:10]; % il punto e virgola evita la rappresentazione dei risultati sulla CW (Command Window)
% Genera i vettori delle funzioni
y1=3*t
y2=t.^2
y3=sin(6.28/5*t)
y4=cos(6.28/5*t)
%Suddivide lo schermo in una "matrice di grafici" di 2 righe per 2 colonne
%Traccia il diagramma di y1 nel grafico n.1
subplot(221),plot(t,y1)
%Dà il titolo all'ultimo grafico tracciato (y1)
title('Grafico di y=3t')
%Dà il nome agli assi dell'ultimo grafico tracciato (y1)
xlabel('t[s]'), ylabel('y(t)')
%Traccia il diagramma di y2 nel grafico n.2
subplot(222),plot(t,y2);
%Dà il titolo all'ultimo grafico tracciato (y2)
title('Grafico di y=t^2')
%Dà il nome agli assi dell'ultimo grafico tracciato (y2)
xlabel('t[s]'),ylabel('y(t)')
%Traccia il diagramma di y3 nel grafico n.3
subplot(223),plot(t,y3)
%Dà il titolo all'ultimo grafico tracciato (y3)
title('Grafico di y=sen(wt)')
%Dà il nome agli assi dell'ultimo grafico tracciato (y3)
xlabel('t[s]'), ylabel('y(t)')
%Traccia il diagramma di y4 nel grafico n.4
subplot(224),plot(t,y4);
%Dà il titolo all'ultimo grafico tracciato (y4)
title('Grafico di cos(wt)')
%Dà il nome agli assi dell'ultimo grafico tracciato (y4)
xlabel('t[s'),ylabel('y(t)')
% Aggiunge la griglia all'ultimo grafico tracciato (y4)
grid
% Introduce la possibilità di zoomare, con l'aiuto del mouse, su tutti i grafici tracciati
zoom on
% Inserisce, con l'aiuto del mouse, una scritta su uno dei grafici tracciati
gtext('Esempio di etichetta')
```

- **% CALCOLA IL DETERMINANTE DI UNA MATRICE E L'INVERSA DELLA STESSA MATRICE**

```
% Costruisce la matrice M di 3 righe per 3 colonne
M = [1 2 3; 5 7 8; 4 2 9]
% Calcola il determinante D della matrice M
D = det(M)
% Calcola la matrice inversa
Mi = inv(M)
```

- **% EFFETTUA ALCUNE VERIFICHE SULLA PROPRIETÀ COMMUTATIVA RIFERITA AL PRODOTTO DI MATRICI**

```
% Costruisce la matrice M1 di 3 righe per 3 colonne
M1 = [1 2 3; 5 7 8; 4 2 9]
% Costruisce la matrice M2 di 3 righe per 3 colonne
M2 = [4 4 1; 2 3 1; 3 6 7]
% Calcola il prodotto M1*M2
P1 = M1*M2
%Calcola il prodotto M2*M1
P2 = M2*M1
% Confrontando i risultati P1 e P2 si può osservare che il prodotto di matrici non gode della
   proprietà commutativa
% Un caso in cui è verificata la proprietà commutativa riguarda il prodotto di una matrice per la
   sua inversa.
% Calcola l'inversa della matrice M1
Mi1 = inv(M1)
% Calcola i due possibili prodotti tra le matrici M1 e Mi1
Q1 = M1*Mi1
Q2 = Mi1*M1
% Confrontando i risultati Q1 e Q2 si può osservare che il prodotto tra la matrice M1 e la sua
inversa Mi1 gode della proprietà commutativa. I due risultati Q1 e Q2 sono infatti uguali tra
loro e coincidono con la matrice identità.
```

- **% SOLUZIONE DI UN SISTEMA DI N EQUAZIONI IN N INCOGNITE**

```
% [A] * [X] = [B]
% [X] = [A]^-1 * [B]
% Matrice dei coefficienti
A = [1 2 -1; 2 1 0; -1 1 2]
% Vettore dei termini noti
B = [1; 0; 0]  % I punti e virgola fanno sì che B sia un vettore colonna.
% Il determinante D viene messo in evidenza al solo scopo di verificare che esso sia diverso da
   0, condizione necessaria perché esista la soluzione
D = det(A)
% Calcola la matrice inversa di A e la moltiplica per il vettore dei termini noti B per ottenere il
   vettore delle soluzioni X
X = inv(A)*B
```

- **% POLI E RESIDUI DI UNA FUNZIONE RAZIONALE FRATTA**

```
% Vettore dei coefficienti del polinomio a numeratore
num = [4 1 -1]
% Vettore dei coefficienti del polinomio a denominatore
den = [1 5 6]
% Calcola:
% i coefficienti dello sviluppo in frazioni parziali (residui)
% i poli della funzione assegnata (poli)
% il termine diretto (k)
[residui, poli, k] = residue(num,den)
% Effettua una verifica calcolando numeratore e denominatore della funzione
% razionale fratta a partire da residui, poli e termine diretto appena
% calcolati.
[num,den] = residue(residui, poli, k)
```

% Osservando i risultati si scopre che numeratore e denominatore così calcolati coincidono con quelli di partenza

- **% ANTITRASFORMATA DELLA FUNZIONE (S+1)/((S+2)*(S+3))**

% Dichiara le variabili s e t (dichiarazione obbligatoria)
Syms s t
% Forma fattorializzata
F = (s+1)/((s+2)*(s+3))
% Calcola l'antitrasformata di F
f = ilaplace(F,s,t)
% Forma rapporto di polinomi
G = (s+1)/(s^2+5*s+6)
% Calcola l'antitrasformata di G
g = ilaplace (G,s,t)

- **% STUDIO NEL DOMINIO DEL TEMPO E IN QUELLO DELLA FREQUENZA DEL SISTEMA CON**

F(s)= 10/(s^2+2*s+5)
% Costruisce la funzione (***solo in questo modo***)
G = tf(10,[1 2 5])
% Apre la finestra di visualizzazione dei grafici
ltiview
% Una volta aperta la finestra grafica, occorre importare la o le funzioni da rappresentare (Menù File → Import…). Poi, con l'aiuto di un menù contestuale (tasto destro del mouse), si possono osservare tutta una serie di figure o parametri.

- **% CONFRONTA LA RISPOSTA AL GRADINO PER UN SISTEMA DEL PRIMO ORDINE, CON E SENZA RETROAZIONE UNITARIA**

% Con la dichiarazione seguente le funzioni saranno generate tutte col metodo ***tf***
s = tf('s')
% Funzione di trasferimento linea diretta
G = 10/(0.01*s + 1)
% Funzione di trasferimento linea di retroazione (unitaria)
H = 1
% Funzione di trasferimento a catena chiusa
W = G/(1+H*G)
% Riduce nella forma minima.
W = mineral(W)
% Apre la finestra di visualizzazione dei grafici
ltiview

- **% CONFRONTA LA RISPOSTA AL GRADINO PER UN SISTEMA DEL SECONDO ORDINE, CON E SENZA RETROAZIONE UNITARIA**

% Con la dichiarazione seguente le funzioni saranno generate tutte col metodo ***tf***
s = tf('s')
% Funzione di trasferimento linea diretta
G = 10/(s^2 + s + 10)
% Funzione di trasferimento linea di retroazione (unitaria)
H = 1
% Funzione di trasferimento a catena chiusa
W = G/(1+H*G)

% Riduce nella forma minima.
W = mineral(W)
% Apre la finestra di visualizzazione dei grafici
ltiview
% In questo caso occorre importare, con l'aiuto del tasto Control, sia la funzione G che la funzione W.

- **% RISPOSTA AL GRADINO UNITARIO DI UN SISTEMA DEL II ORDINE, PER VARI VALORI DI ZITA**

% zita = 2
F1 = tf(10, [1 4 1])
% zita = 1
F2 = tf(10, [1 2 1])
% zita = 0,7
F3 = tf(10, [1 1.4 1])
% zita = 0,2
F4 = tf(10, [1 0.4 1])
% zita = 0,1
F5 = tf(10, [1 0.2 1])
% zita = 0
F6 = tf(10, [1 0 1])
% Apre la finestra di visualizzazione dei grafici
ltiview
% Una volta aperta la pagina grafica, importare con l'aiuto del tasto Control le 6 funzioni. Il grafico che si ottiene mostra come la risposta, al variare di zita, passi da un andamento esponenziale (zita = 2 --> 1) ad un andamento oscillante smorzato con sovraelongazioni crescenti (zita = 1 --> 0,1), fino ad un andamento oscillante non smorzato (zita = 0).

- **% PARAMETRI DELLA RISPOSTA AL GRADINO PER UN SISTEMA DEL SECONDO ORDINE.**

% I parametri da considerare sono:
% a) Sovraelongazione $M\% = \frac{V_{max} - V_{regime}}{V_{regime}} * 100 = 100 * e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}}$
% b) Pulsazione smorzata $\omega = \omega_n * \sqrt{1-\zeta^2}$
% c) Tempo della sovraelongazione $T_p = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}} = \frac{\pi}{\omega}$
% d) Tempo di assestamento $T_s \cong \frac{3}{\omega_n \zeta}$
% e) Tempo di ritardo $T_d \cong \frac{1+0{,}7\zeta}{\omega_n}$
% f) Tempo di salita $T_r \cong \frac{1+1{,}1\zeta+1{,}4\zeta^2}{\omega_n}$
% La funzione di trasferimento non deve avere zeri.
% ***F(s) = 1/(s^2 + 0,4s + 1)***
% $\zeta = 0{,}2$
% $\omega_n = 1$
%

F = tf([1],[1 0.4 1])
% Apre la finestra di visualizzazione dei grafici e importa la funzione F
ltiview(F)
% Dopo l'apertura della finestra grafica, col tasto destro del mouse attivare l'indicazione dei parametri caratteristici della risposta. Per la lettura occorre passare sulla figura il puntatore del mouse, nei punti che interessano, e leggere le informazioni che compaiono.
% Solo per il tempo di ritardo (Td), non previsto, effettuare una misura manuale cliccando e spostando il cursore del mouse lungo la figura, fino a leggere una ampiezza pari a 0,5; in corrispondenza del punto trovato leggere Td.

% Calcolo teorico dei parametri
zita = 0.2
omegan = 1
omega = omegan*sqrt(1-zita^2)
M = 100*exp(-pi*zita/sqrt(1-zita^2))
Tp = pi/omega
Ts = 3/(omegan*zita)
Td = (1+0.7*zita)/omegan
Tr = (1+1.1*zita+1.4*zita^2)/omegan

(******* non ricopiare sull'M-file **********)***

Si ottengono i seguenti risultati:

	Valori misurati	**Valori calcolati**
M%	**52,7%**	**52,7%**
T_p	**3,2 s**	**3,2 s**
T_s	**19,6 s (±2%)**	**15 s (±5%)**
T_d	**1,13 s**	**1,14 s**
T_r	**1,21 s**	**1,28 s**
ω	**0,979 rad/s**	**0,980 rad/s**

- **% RISPOSTA AL GRADINO, ALLA RAMPA LINEARE E ALLA RAMPA PARABOLICA.**

% La funzione "***ltiview***", per poter visualizzare la risposta al gradino, moltiplica automaticamente la funzione di trasferimento ***G(s)*** per ***1/s***, che rappresenta l'ingresso a gradino unitario; e fa questo in ogni caso. Se moltiplichiamo preventivamente G(s) per ***1/s***, la rappresentazione della risposta al gradino di ltiview si trasformerà nella rappresentazione della risposta alla rampa lineare, perchè la G(s) risulterà moltiplicata, complessivamente, per ***$1/s^2$***, che è la trasformata della rampa lineare unitaria.
G = tf(1,[1 0.4 1])
% Moltiplica G(s) per ***1/s***
Grl = G*tf(1,[1 0])
% Allo stesso modo, se moltiplichiamo G(s) per ***$2/s^2$***, all'attivazione di ***ltiview*** (risposta al gradino) essa sarà moltiplicata complessivamente per ***$2/s^2$***, che è la trasformata di una rampa parabolica unitaria.
% Moltiplica G(s) per ***$2/s^2$***.
Grp = G*tf(2,[1 0 0])

% Apre la finestra di visualizzazione dei grafici
ltiview(G, Grl, Grp)
*% Scegliere la visualizzazione della risposta al gradino. Per una visualizzazione ottimale delle tre figure, attivare il menù contestuale (tasto destro del mouse sul grafico), poi scegliere "**Properties**" --> "**Limits**" --> **Deselezionare** le caselle dell'autoscala per gli assi **X** e **Y** --> Impostare **X = 0 .. 20** e **Y = 0 ..20**.*

- **% TRACCIA I DIAGRAMMI DI BODE PER LA LINEA DIRETTA (G), PER LA LINEA DI RETROAZIONE (H), PER LA FUNZIONE A CICLO APERTO (F) E PER LA FUNZIONE A CICLO CHIUSO (W).**

% Funzione della linea diretta
G = tf(1000, [3 4 1])
% Funzione della linea di retroazione
H = tf(1,[1 10])
% Funzione a ciclo aperto
F = G * H
% Riduce ***F*** nella forma minima
F = minreal(F)
% Funzione a ciclo chiuso
W = G/(1+F)
% riduce ***W*** nella forma minima
W = minreal(W)
% Traccia i diagrammi di Bode
Bode(G, H, F,W)
*Selezionare "**Systems**" dal menù contestuale per associare i grafici alle funzioni.*
% Determina ***modulo (in assoluto)*** e ***fase*** di ***W*** in corrispondenza della pulsazione ***100 rad/s***.
[m, f] = Bode(W, 100)
% Calcola ***m*** in ***dB***.
mdB = 20*Log10(m)

- **% TRACCIA I DIAGRAMMI DI BODE IMPONENDO LO STILE PER I SINGOLI GRAFICI.**

% Funzione della linea diretta
G = tf(1000, [3 4 1])
% Funzione della linea di retroazione
H = tf(1,[1 10])
% Funzione a ciclo aperto
F = G * H
% Riduce F nella forma minima.
F = minreal(F)
% Funzione a ciclo chiuso
W = G/(1+F)
% Riduce W nella forma minima.
W = minreal(W)
% Traccia i diagrammi di Bode imponendo lo stile per i singoli grafici.
Bode(G,'r',H,'y--',F,'b-.',W,'g.')
% ***G*** è una curva continua di colore ***rosso*** (red);
% ***H*** è una curva tratteggiata di colore ***giallo*** (yellow);
% ***F*** è una curva tratto e punto di colore ***blue*** (blue);
% ***W*** è una curva tracciata solo con markers (**.**) e di colore ***verde*** (green).

- **% TRACCIA I DIAGRAMMI DI NYQUIST PER LA LINEA DIRETTA (G), PER LA LINEA DI RETROAZIONE (H), PER LA FUNZIONE A CICLO APERTO (F) E PER LA FUNZIONE A CICLO CHIUSO (W).**

```
% Funzione della linea diretta
G = tf(1000, [3 4 1])
% Funzione della linea di retroazione
H = tf(1,[1 10])
% Funzione a ciclo aperto
F = G * H
% Funzione a ciclo chiuso
W = G/(1+F)
% Riduce W nella forma minima.
W = minreal(W)
% Traccia i diagrammi di Nyquist
nyquist(G,H,F,W)
% Selezionare "Systems" dal menù contestuale per associare i grafici alle funzioni.
% Determina parte reale e parte immaginaria di W in corrispondenza della pulsazione 100 rad/s.
 [r, i] = nyquist(W, 100)
```

- **% TRACCIA I DIAGRAMMI DI NYQUIST IMPONENDO LO STILE PER I SINGOLI GRAFICI.**

```
% Funzione della linea diretta
G = tf(1000, [3 4 1])
% Funzione della linea di retroazione
H = tf(1,[1 10])
% Funzione a ciclo aperto
F = G * H
% Riduce F nella forma minima.
F = minreal(F)
% Funzione a ciclo chiuso
W = G/(1+F)
% Riduce W nella forma minima.
W = minreal(W)
% Traccia i diagrammi di Bode imponendo lo stile per i singoli grafici.
nyquist(G,'r',H,'y--',F,'b-.',W,'g.')
% G è una curva continua di colore rosso (red);
% H è una curva tratteggiata di colore giallo (yellow);
% F è una curva tratto e punto di colore blue (blue);
% W è una curva tracciata solo con markers (.) e di colore verde (green).
```

- **% TRACCIA DIAGRAMMI DI BODE E DI NYQUIST UTILIZZANDO DUE FINESTRE DIVERSE.**

```
% Crea la funzione F(s)
F = tf(10,[1 0.6 1])
% Traccia i diagrammi di Bode nella Fig. 1
bode(F)
% Apre una nuova finestra (Fig. 2)
figure
% Traccia il diagramma di Nyquist nella Fig. 2
nyquist(F)
```