CONTROL SYSTEM TOOLBOX per MATLAB 5.2

(a cura di Bettini Francesca)

E` un insieme di funzioni per l'analisi di sistemi dinamici (tipicamente lineari tempo invarianti o LTI) e per la sintesi di controllori (in particolare a retroazione). All'interno di questa libreria le funzioni sono poi suddivise in gruppi (vedi `help control`), fra cui noi andremo ad analizzare in particolare quelli preposti a:
- creazione modelli LTI;
- conversioni;
- connessione di sistemi;
- analisi nel dominio del tempo;
- analisi nel dominio della frequenza;
- sintesi del controllore.

Si ha inoltre la possibilità di interagire con Simulink.

Creazione modelli LTI

I sistemi dinamici lineari tempo invarianti possono essere rappresentati mediante rappresentazione in spazio di stato, in termini di funzioni di trasferimento o di zeri/poli.

Modello in spazio di stato:
Sono note le equazioni differenziali del sistema in funzione delle variabili di stato, cioè, nel caso di sistemi a tempo continuo, si ha

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

¿
¿¿¿
¿

che diventano analoghe equazioni alle differenze nel caso di sistemi a tempo discreto.
Un sistema dinamico e` quindi descritto mediante le matrici A, B, C, D le cui dimensioni determinano il numero di stati, ingressi e uscite.
Per creare in MATLAB un modello in spazio di stato è disponibile la funzione `ss`

`sys=ss(A,B,C,D)`	crea il modello in spazio di stato scritto prima
`sys=ss(A,B,C,D,T)`	crea un modello in spazio di stato a tempo discreto con periodo di campionamento T

se prima sono state introdotte le matrici A, B, C, D del sistema

N.B. per visualizzare un modello introdotto in MATLAB si può usare la funzione `printsys`.

Modello in termini di funzione di trasferimento

La relazione ingresso uscita del sistema è cioè data in termine della funzione di trasferimento, è quindi nota W(s)=Y(s)/U(s).

Per rappresentare un sistema tramite funzione di trasferimento in MATLAB è disponibile la funzione `tf`. Si hanno diversi casi a seconda del numero di ingressi e di uscite del sistema in esame.

Nel caso di sistemi SISO la funzione di trasferimento è semplicemente il rapporto fra il polinomio numeratore e quello del denominatore. I polinomi in MATLAB vengono rappresentati da vettori contenenti i coefficienti in ordine decrescente, cioè

s^3+2s+3 [1 0 2 3]

N.B. bisogna sempre ricordarsi di mettere 0 quando manca un termine.

Quindi per creare il modello avente funzione di trasferimento

$$W(s)=\frac{s^2+1}{s^3-2s^2+3}$$

sono sufficienti le seguenti istruzioni:

```
» num=[1 0 1];            (assegno il numeratore)
» den=[1 -2 0 3];         (assegno il denominatore)
» sys=tf(num,den)

Transfer function:
    s^2 + 1
---------------
s^3 - 2 s^2 + 3
»
```

Nel caso di sistemi MIMO la funzione di trasferimento è una matrice con m (numero uscite) righe e n (numero ingressi) colonne. In MATLAB ciò viene tradotto usando 2 celle di n×m vettori, uno con tutti i numeratori delle funzioni di trasferimento e l'altro con i denominatori, quindi la coppia num{i,j}, den{i,j} specifica la funzione di trasferimento fra l'ingresso j e l'uscita i.

Cioè ad esempio un sistema con un ingresso e due uscite avente funzione di trasferimento:

$$W(s)=\begin{bmatrix}\frac{2s^2+12}{s^3+11s+6}\\ \frac{5s-6}{6s^2+s+1}\end{bmatrix}$$

in MATLAB viene costruito tramite:

```
» num={[2 12];[5 -6]};
» den={[1 0 11 6];[6 1 1]};
» sys=tf(N,d)
Transfer function from input 1 to output...
        1
 #1:  -----
      s + 1

         1
 #2:  -------
```

```
      2 s + 1

Transfer function from input 2 to output...
        1
 #1:  -----
      s + 2

       2
 #2:  ---
      2 s
»
```

Per sistemi discreti la creazione del modello a partire dalla funzione di trasferimento è analoga solo che nei vettori che rappresentano il numeratore e il denominatore della f.d.t si pongono i coefficienti, sempre in ordine decrescente, di z e si deve specificare il periodo di campionamento.
Es.

```
» num=[1 2];
» den=[1 0 1];
» T=1;
» sys=tf(num,den,T)

Transfer function:
 z + 2
-------
z^2 + 1

Sampling time: 1
»
```

<u>Modello in termini di guadagno, zeri e poli</u>
Si consideri per semplicità un sistema SISO. Se sono noti gli zeri, i poli e il guadagno della funzione di trasferimento del sistema, cioè si ha

$$W(s)=k\frac{\prod_{i=1}^{m}(s-z_i)}{\prod_{i=1}^{n}(s-p_i)}$$

per costruire l'equivalente modello in MATLAB è sufficiente scrivere in un vettore gli zeri, in un altro i poli e specificare il valore del guadagno e quindi usare la funzione `zpk`
Es. Per creare il modello del sistema avente uno zero in 1, due poli (uno in -1 e l'altro in 0) e guadagno 10 sono sufficienti le istruzioni:

```
» zeri=1;
» poli=[-1 0];
» k=10;
» sys=zpk(z,p,k)

Zero/pole/gain:
10 (s-1)
--------
```

```
s (s+1)

»
```

N.B. Se il sistema non ha zeri si pone `zeri=[]`

Per sistemi MIMO valgono considerazioni analoghe al caso precedente.

N.B. Per l'antitrasformazione è utile ricondurre la f.d.t del sistema da rapporto di polinomi a somma di fratti semplici, e quindi si devono calcolare i residui. MATLAB fornisce la funzione `residue` che, dati i polinomi numeratore e denominatore, sempre come vettori dei coefficienti in potenze decrescenti di s, restituisce i residui, i poli e il termine costante, purchè i poli siano tutti semplici (vedi `help residue`).

N.B. Per avere maggiori dettagli sulle proprietà dei modelli LTI creati con MATLAB e sulla terminologia presente negli help vedi `help ltiprops`

Conversioni.

Da una rappresentazione ad un'altra
Le funzioni introdotte prima (`ss, tf, zpk`) servono anche per convertire un sistema da una forma ad un'altra, cioè se nella variabile sys è ad esempio memorizzato un sistema mediante spazio di stato, il comando `sys1=tf(sys)` crea in sys1 l'equivalente sistema in termini di funzione di trasferimento.
Esistono inoltre le seguenti funzioni per le conversioni:

`ss2tf, tf2ss`	spazio di stato <-> f.d.t.
`ss2zp, zp2ss`	spazio di stato <-> zeri/poli/guadagno
`tf2zp, zp2tf`	zeri/poli/guadagno <-> f.d.t.

Da discreto a continuo e viceversa
Funzioni di conversione continuo/discreto:

`c2d, c2dm`	conversione da continuo a discreto
`d2c, d2cm`	conversione da discreto a continuo
`d2d`	ricampiona un sistema LTI discreto, o introduce un ritardo in ingresso

Connessione di sistemi

`append`	Raggruppa sistemi LTI, accodando ingressi e uscite.
`feedback`	Crea la connessione in retroazione di due sistemi. Per default si crea retroazione negativa
`parallel`	Crea la connessione in parallelo di due sistemi LTI.
`series`	Crea la connessione in serie di due sistemi LTI.
`connect`	Ricava il modello in spazio di stato per il diagramma a blocchi dell'interconnessione
`ssdelete`	Elimina ingressi, uscite o stati da un sistema rappresentato in spazio di stato.
`ssselect`	Seleziona un sottosistema da un sistema più grande.

Analisi nel dominio del tempo

Funzioni per l'analisi del comportamento nel dominio del tempo di sistemi LTI a cui si applicano ingressi canonici o di tipo generico.

`impulse, dimpulse`	Risposta all'impulso di un sistema LTI continuo e discreto, rispettivamente.
`initial, dinitial`	Evoluzione libera di un sistema LTI continuo e discreto, rispettivamente.
`step, dstep`	Risposta ad un gradino di un sistema LTI continuo e discreto, rispettivamente.
`lsim, dlsim`	Risposta ad un ingresso arbitrario di un sistema LTI continuo e discreto, rispettivamente.
`ltiview`	Apre un'interfaccia grafica per l'analisi della risposta nel tempo.
`gensig`	Generatore di segnali periodici per la simulazione della risposta nel tempo con la funzione `lsim`.
`stepfun`	Crea la funzione gradino.

Le funzioni accettano in ingresso sistemi LTI in una qualsiasi delle rappresentazioni.
Hanno diverse sintassi possibili. In particolare se richiamate senza assegnare variabili di uscita (Es. `step(sys)`) generano il plot, altrimenti possono ritornare i valori numerici in un vettore.

Analisi nel dominio della frequenza

Funzioni per l'analisi del comportamento nel dominio della frequenza di sistemi LTI.

`bode,dbode`	Risposta in frequenza per diagramma di Bode (rispettivamente per sistemi continui e discreti).
`nyquist,dnyquist`	Risposta in frequenza per diagramma di Nyquist.
`nichols,dnichols`	Risposta in frequenza per diagramma di Nichols.
`ngrid`	Disegna una griglia per il diagramma di Nichols.

`freqresp`	Risposta in frequenza di un sistema LTI.
`evalfr`	Valuta la risposta in frequenza ad una data frequenza.
`margin`	Ritorna il margine di fase e di ampiezza e la frequenza di attraversamento.

Esempio: diagramma di Bode in Hertz

```
» num=[1];
» den=[1 1];
» sys=tf(num,den);
» [mag,phase,w]=bode(sys);     (se richiamato senza uscite disegna subito il diagramma di Bode)
» subplot(211)
» semilogx(w/(2*pi),20*log10(mag))
» xlabel('Frequency (Hz)')
» ylabel('Gain (dB)')
» subplot(212)
» semilogx(w/(2*pi),phase)
» xlabel('Frequency (Hz)')
» ylabel('Phase (deg)')
```

Progetto o sintesi del controllore

Funzioni per il progetto di controllori in retroazione.

`pzmap`	Mappa di zeri e poli di un sistema lineare.
`rlocus`	Traccia il luogo delle radici.
`rlocfind`	Ricava il luogo delle radici per determinati valori delle radici.
`sgrid`	Disegna una griglia nel piano s per il luogo delle radici e la mappa zeri/poli.
`zgrid`	Disegna una griglia nel piano z per il luogo delle radici e la mappa zeri/poli.
`damp`	Frequenza naturale e smorzamento dei poli di un sistema