Control System Toolbox

Matlab 5.3

Generalità

- il Control System Toolbox offre una serie di strumenti per l'analisi dei sistemi dinamici
- i modelli possono essere rappresentati in varie forme:
  1. State-Space
  2. Transfer-Function
  3. Zero-Pole-Gain

Introduzione ai modelli di sistemi LTI

Verrà ora preso in esame il caso dei sistemi lineari a tempo invariante (LTI).
Il Control System Toolbox ne consente la rappresentazione nelle seguenti forme:

- **modelli State-Space (SS)**
  Un modello nello spazio degli stati viene identificato dalle quattro matrici *(A, B, C, D)*

$$\begin{aligned} \frac{dx}{dt} &= Ax + Bu \\ y &= Cx + Du \end{aligned}$$

  dove *A, B, C* e *D* sono matrici di dimensioni opportune, *x* è il vettore di stato, mentre *u* ed *y* sono rispettivamente i vettori degli ingressi e delle uscite del sistema.

  Ad esempio, un sistema del secondo ordine caratterizzato da una coppia di poli complessi coniugati con pulsazione naturale $\omega_n = 1.5$ *rad/s* e rapporto di smorzamento $\xi = 0.2$ si può rappresentare come

$$\begin{aligned} \dot{x}_1(t) &= x_2(t) \\ \dot{x}_2(t) &= -\omega_n{}^2 x_1(t) - 2\xi\omega_n x_2(t) + \omega_n{}^2 u(t) \\ y &= x_1(t) \end{aligned}$$

  per cui le matrici *A, B, C* e *D* sono

$$A = \begin{bmatrix} 0 & 1 \\ -\omega_n{}^2 & -2\xi\omega_n \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ \omega_n{}^2 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D = 0$$

- **modelli Transfer Function (TF)**
  La forma transfer function permette di rappresentare un modello tramite una funzione di trasferimento *G(s)* definita dai polinomi *NUM(s)* e *DEN(S)*

$$G(s) = C(sI - A)^{-1} B + D = \frac{NUM(s)}{DEN(s)}$$

Nel caso dell'esempio precedente si ha

$$G(s) = \frac{2.25}{s^2 + 0.6s + 2.25}$$

- **modelli Zero-Pole-Gain (ZPK)**
  Si tratta della forma fattorizzata della funzione di trasferimento

$$G(s) = k\frac{(s - z_1)(s - z_2)...(s - z_m)}{(s - p_1)(s - p_2)...(s - p_n)}$$

  Nell'esempio preso in esame si ha

$$G(s) = 2.25\frac{1}{(s + 0.3 + 1.46i)(s + 0.3 - 1.46i)}$$

Rappresentazione dei sistemi LTI in Matlab

La versione 5.3 di Matlab consente di rappresentare un sistema LTI come un oggetto unico, a partire da una delle rappresentazioni precedentemente descritte. Di seguito si farà sempre riferimento all'esempio finora utilizzato.

- **modelli State-Space (SS)**
  innanzitutto vanno definite in Matlab le quattro matrici *A, B, C* e *D*

```
>> wn = 1.5;
>> csi = 0.2;
>> A = [0 1; -wn^2 –2*csi*wn];
>> B = [0; wn^2];
>> C = [1 0];
>> D = 0;
```

  (`wn` e `csi` sono la pulsazione naturale ω_n ed il rapporto di smorzamento ξ rispettivamente) successivamente si utilizza il commando **ss**

```
>> sys_ss = ss(A,B,C,D);
```

- **modelli Transfer Function (TF)**
  si definiscono i due polinomi *NUM(s)* e *DEN(s)*

```
>> NUM = 2.25;
>> DEN = [1 0.6 2.25];
```

  e si utilizza il comando **tf**

```
>> sys_tf = tf(NUM,DEN);
```

- **modelli Zero-Pole-Gain (ZPK)**
  si definiscono il guadagno *k* ed i vettori degli zeri e dei poli, *z* e *p* rispettivamente

```
>> k = 2.25;
>> z = [];
>> p = [-0.3+1.4697i; -0.3-1.4697i];
```

  in questo caso il comando da utilizzare è **zpk**

```
>> sys_zpk = zpk(z,p,k);
```

Conversioni fra modelli

Le funzioni finora presentate (ss, tf, zpk) sono anche in grado di operare una conversione tra le diverse rappresentazioni dei modelli. Ad esempio, la sintassi seguente consente di ottenere una rappresentazione tramite funzione di trasferimento di un modello state-space:

```
>> sys_tf = tf(sys_ss);
```

Le funzioni **ssdata**, **tfdata** e **zpkdata** consentono invece di estrarre da un modello LTI qualsiasi (`sys`) i dati caratteristici di una particolare rappresentazione

```
>> [A,B,C,D] = ssdata(sys)
>> [NUM,DEN] = tfdata(sys,'v')
>> [Z,P,K] = zpkdata(sys,'v')
```

dove il paramentro 'v', da utilizzarsi nel caso di sistemi SISO, specifica di restituire il risultato in forma vettoriale.

Considerando sempre un sistema LTI (`sys`) definito attraverso una qualunque delle rappresentazioni possibili, la funzione **pole** ne calcola i poli

```
>> P = pole(sys)
```

la funzione **pzmap** ne traccia graficamente la mappa poli-zeri

`>> pzmap(sys)`	disegna la mappa poli-zeri del sistema
`>> pzmap(sys1,sys2,...)`	disegna sullo stesso grafico le mappe poli-zeri di più sistemi LTI permettendone il confronto
`>> [P,Z]=pzmap(sys)`	calcola i poli e gli zeri del sistema e li salva nei vettori `P` e `Z` rispettivamente

mentre la funzione **dcgain** ne calcola il guadagno statico

```
>> dcgain(sys)
```

Analisi dei sistemi LTI

A partire dalla versione 5.3, Matlab mette a disposizione un potente strumento per l'analisi dei sistemi LTI: **ltiview**. L'interfaccia grafica permette di selezionare diverse visualizzazioni (risposta a scalino, risposta ad impulso, diagrammi di Bode e Nyquist, ecc…), a cui si accede tramite il menu tasto destro del mouse.

Per analisi più specifiche, è possibile utilizzare funzioni dedicate, elencate di seguito (negli esempi si consideri l'oggetto `sys` come modello di un sistema LTI rappresentato in una qualsiasi delle forme precedentemente descritte):

- **step** – risposta a scalino

`>> step(sys)`	traccia la risposta a scalino del sistema
`>> step(sys,Tend)`	traccia la risposta a scalino fino all'istante di tempo finale specificato (`Tend`)
`>> step(sys,t)`	traccia la risposta a scalino utilizzando il vettore degli istanti di tempo specificato (`t`)
`>> step(sys1,sys2,…,t)`	traccia le risposte a scalino di più sistemi mettendole a confronto; il parametro t è opzionale
`>> Y=step(sys,t)` `>> [Y,t]=step(sys)`	calcola la risposta a scalino la salva nel vettore `Y`; il vettore degli istanti di tempo `t` può essere specificato come parametro o come output

- **impulse** – risposta ad impulso

`>> impulse(sys)`	traccia la risposta ad impulso del sistema
`>> impulse(sys,Tend)`	traccia la risposta ad impulso fino all'istante di tempo finale specificato (Tend)
`>> impulse(sys,t)`	traccia la risposta ad impulso utilizzando il vettore degli istanti di tempo specificato (t)
`>> impulse(sys1,sys2,…,t)`	traccia le risposte ad impulso di più sistemi mettendole a confronto; il parametro t è opzionale
`>> Y=impulse(sys,t)` `>> [Y,t]=impulse(sys)`	calcola la risposta ad impulso la salva nel vettore `Y`; il vettore degli istanti di tempo `t` può essere specificato come parametro o come output

- **initial** – movimento libero

`>> initial(sys,X0)`	traccia l'andamento del movimento libero dello stato a partire dalla condizione iniziale `X0`
`>> initial(sys,X0,Tend)`	traccia l'andamento del movimento libero dello stato fino all'istante di tempo finale specificato (Tend)
`>> initial(sys,X0,t)`	traccia l'andamento del movimento libero dello stato utilizzando il vettore degli istanti di tempo specificato (t)
`>> impulse(sys1,sys2,…,X0,t)`	traccia l'andamento del movimento libero dello

stato di più sistemi mettendole a confronto; il parametro t è opzionale

```
>> [Y,t,X]=impulse(sys,X0)
```
calcola l'andamento del movimento libero dello stato e salva l'uscita del sistema nel vettore `Y`; il vettore degli istanti di tempo in `t`; la traiettoria dello stato in `X`

- **bode** – diagramma di Bode

```
>> bode(sys)
```
traccia il diagramma di Bode (margine e fase) del sistema

```
>> bode(sys,{wmin,wmax})
```
traccia il diagramma di Bode del sistema nell'intervallo di frequenze limitato dai valori `wmin` e `wmax`

```
>> bode(sys,w)
```
traccia il diagramma di Bode utilizzando il vettore delle frequenze specificato (`w`)

```
>> bode(sys1,sys2,…,w)
```
traccia i diagrammi di Bode di più sistemi mettendoli a confronto; il parametro `w` è opzionale

```
>> [MAG,PHASE]=bode(sys,w)
>> [MAG,PHASE,w]=bode(sys)
```
calcola il diagramma di Bode e salva in `MAG` e `PHASE` i vettori dei guadagni e degli sfasamenti della risposta in frequenza rispettivamente; il vettore delle frequenze `w` può essere specificato come argomento oppure richiesto come output

- **margin** – margine di fase

```
>> margin(sys)
```
traccia il diagramma di Bode del sistema, indicando i margini di fase e di guadagno con una linea verticale

```
>> [Gm,Pm,Wg,Wp]=margin(SYS)
```
traccia il diagramma di Nyquist utilizzando il vettore delle frequenze specificato (`w`)

- **nyquist** – diagramma di Nyquist

```
>> nyquist(sys)
```
traccia il diagramma di Nyquist del sistema

```
>> nyquist(sys,{wmin,wmax})
```
traccia il diagramma di Nyquist del sistema nell'intervallo di frequenze limitato dai valori `wmin` e `wmax`

```
>> nyquist(sys,w)
```
traccia il diagramma di Nyquist utilizzando il vettore delle frequenze specificato (`w`)

```
>> nyquist(sys1,sys2,…,w)
```
traccia i diagrammi di Nyquist di più sistemi mettendoli a confronto; il parametro `w` è opzionale

```
>> [RE,IM]=nyquist(sys,w)
>> [RE,IM,w]=nyquist(sys)
```
calcola il diagramma di Nyquist e salva in `RE` e `IM` i vettori della parte reale e della parte immaginaria della risposta in frequenza rispettivamente; il vettore delle frequenze `w` può essere specificato come argomento oppure richiesto come output

- **rlocus** – luogo delle radici

```
>> rlocus(sys)
```
traccia il diagramma di Nyquist del sistema

```
>> rlocus(sys,k)
```
traccia il diagramma di Nyquist utilizzando il vettore delle frequenze specificato (`w`)

```
>> rlocus(sys1,sys2,…)
```
traccia i diagrammi di Nyquist di più sistemi mettendoli a confronto; il parametro `w` è opzionale

```
>> R= rlocus(sys,k)
>> [RE,IM,w]= rlocus(sys)
```
calcola il diagramma di Nyquist e salva in `RE` e `IM` i vettori della parte reale e della parte immaginaria della risposta in frequenza rispettivamente; il vettore delle frequenze `w` può essere specificato come argomento oppure richiesto come output