Sistema massa-molla-smorzatore

Consideriamo il sistema massa-molla-smorzatore a pagina 2-7 delle dispense, descritto dalle equazioni:

$$\begin{aligned} v(t) &= \dot{s}(t) \\ m\dot{v}(t) &= u(t) - \beta v(t) - k s(t) \end{aligned} \tag{1}$$

Vogliamo descrivere il sistema nella forma:

$$\begin{aligned} \dot{x}(t) &= A\,x(t) + B\,u(t) \\ y(t) &= C\,x(t) + D\,u(t) \end{aligned} \tag{2}$$

Definendo $x(t) \triangleq \begin{bmatrix} s(t) \\ v(t) \end{bmatrix}$ e $y(t) \triangleq s(t)$, dalle (1) otteniamo:

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{\beta}{m} \end{bmatrix} \quad , \quad B = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} \quad , \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad , \quad D = 0\,.$$

Il seguente script MATLAB simula il sistema massa-molla-smorzatore per diverse condizioni iniziali $x(0)$ e diversi ingressi $u(t)$, graficandone l'andamento dell'uscita. Salvare il file con estensione `.m`, per esempio `MassaMolla.m`; quindi, per eseguire lo script, digitare `MassaMolla` al prompt dei comandi.

```
% Definizione delle costanti del modello
m=100;
k=10;
beta=25;

% Definizione delle matrici del sistema
A=[ 0 1 ; -k/m -beta/m];
B=[ 0 ; 1/m ];
C=[ 1 0 ];
D=0;

% Creazione del sistema
S=ss(A,B,C,D);

% Calcolo degli autovalori del sistema
pole(S)

% Calcolo del guadagno in continua del sistema
dcgain(S)

% Prima simulazione del sistema
X0=[ 2 ; 0 ];
T=0:0.1:50;
U=zeros(length(T),1);
```

```
Y=lsim(S,U,T,X0);
figure(1)
plot(T,Y)

% Seconda simulazione del sistema
X0=[ 0 ; 0 ];
T=0:0.1:50;
umax=3;
U=umax*ones(length(T),1);
Y=lsim(S,U,T,X0);
figure(2)
plot(T,Y)

% Terza simulazione del sistema
X0=[ 0 ; 0 ];
T=0:0.1:80;
umax=5;
omega=0.8;
U=umax*sin(omega*T');
Y=lsim(S,U,T,X0);
figure(3)
plot(T,Y)
```

La funzione `ss` crea nel workspace di Matlab l'oggetto *sistema lineare a tempo continuo* descritto dalle equazioni (2). La sintassi della funzione è la seguente:

`S=ss(A,B,C,D)`.

La funzione `lsim` simula la risposta di un sistema lineare a ingressi arbitrari. Digitando `help lsim` è possibile avere la descrizione di tutti i possibili utilizzi della funzione. Nel programma qui presentato essa viene utilizzata secondo la sintassi:

`Y=lsim(S,U,T,X0)`,

dove `S` è un sistema lineare (definito per esempio con `ss`), `U` è la matrice degli ingressi, `T` è il vettore tempo, e `X0` è la condizione iniziale. Il vettore `T` contiene gli istanti di tempo in cui viene valutata la risposta del sistema; tipicamente esso consiste di campioni equidistanti nel tempo. Per esempio definendo:

`T=0:0.01:5;`

`lsim` simula il sistema per 5 secondi, valutandone la risposta ogni 0.01 secondi. La matrice `U` deve avere tante colonne quanti sono gli ingressi (nel nostro caso una, perché abbiamo un solo ingresso), e tante righe quanto `length(T)` (cioè la lunghezza del vettore `T`). Infatti, la colonna i-esima di `U` consiste dei valori dell'ingresso i-esimo applicati agli istanti di tempo definiti in `T`. La funzione `lsim` restituisce nella matrice `Y` (che ha tante colonne quante sono le uscite, nel nostro caso solo una) i valori delle

uscite del sistema calcolati agli istanti di tempo definiti in `T`.
Nel nostro programma, la prima simulazione viene svolta con condizione iniziale non nulla e ingresso nullo. Quest'ultimo viene definito dalla linea di comando:

```
U=zeros(length(T),1);
```

che genera un vettore colonna di zeri di lunghezza `length(T)`. La seconda simulazione viene invece svolta con condizione iniziale nulla e ingresso costante. Quest'ultimo viene definito dalla linea di comando:

```
U=umax*ones(length(T),1);
```

che genera un vettore colonna di lunghezza `length(T)` in cui ciascun elemento ha valore `umax`. Infine la terza simulazione viene svolta con condizione iniziale nulla e ingresso sinusoidale. Quest'ultimo viene definito dalla linea di comando:

```
U=umax*sin(omega*T');
```

che genera un vettore colonna il cui elemento i-esimo è `umax*sin(omega*T(i))`. Si osservi che `umax` è l'ampiezza della sinusoide, mentre `omega` ne è la pulsazione.
Per visualizzare la risposta del sistema viene utilizzata la funzione:

```
plot(T,Y).
```

Il vettore `T` rappresenta qui il vettore delle ascisse, mentre `Y` rappresenta il vettore delle corrispondenti ordinate: `Y(i)` è infatti il valore dell'uscita del sistema all'istante `T(i)`. La funzione `figure(H)`, dove `H` è un intero positivo, seleziona la figura `H` come figura corrente.

Esercizi

1. Simulare il sistema per diversi valori di m, k e β (positivi!), osservando come cambia l'andamento delle varie risposte; trovare una giustificazione fisica agli andamenti osservati. Verificare che gli autovalori del sistema hanno sempre parte reale negativa
   Osservazione. Cambiando i valori delle costanti del modello, potrebbe risultare necessario adeguare gli intervalli di tempo (quindi il vettore `T`...) in cui vengono calcolate e graficate le risposte del sistema.

2. Provare a variare la condizione iniziale `X(0)` dello stato e i parametri dell'ingresso (cioè `umax` e/o `omega`) per le diverse simulazioni. In particolare, nel caso di ingresso costante di ampiezza `umax` (seconda simulazione), verificare in Figura 2 che il valore di regime è sempre pari al prodotto del guadagno in continua del sistema per `umax`.