Laboratorio 1 per Fondamenti di Automatica – Ingegneria Fisica

Sistemi dinamici lineari con Matlab

Tracce di possibili soluzioni

Esercizio 1

Per studiare la stabilità del sistema e per valutarne il comportamento qualitativo, occorre conoscere gli autovalori della matrice A. Il sistema è a tempo continuo; la stabilità asintotica è quindi garantita se e solo se tutti gli autovalori di A hanno parte reale negativa.
Dunque, si può scrivere:

```
>> A=[-1 -1 2;
0 -2 10;
0 -10 -2];                                    % matrice A
```

Calcolo gli autovalori e la loro parte reale

```
>> autoval=eig(A)                             % autovalori di A

autoval =

-1.0000
-2.0000 +10.0000i
-2.0000 -10.0000i

>> REval= real(autoval)

REval =

-1
-2
-2
```

Dato che tutti gli autovalori hanno parte reale negativa, si conclude che il sistema è asintoticamente stabile.
Inoltre la matrice A ha due autovalori complessi coniugati; pertanto, ci si può aspettare una risposta allo scalino che presenta oscillazioni smorzate.

La costante di tempo dominante del sistema è pari a $T_D = -\frac{1}{\Re(\lambda)_{MAX}}$, mentre il tempo di risposta è pari a $T_R = 5T_D$. Pertanto $T_D = 1$ e $T_R = 5$.

Per calcolare l'equilibrio del sistema, occorre imporre

$Ax + bu = 0$

da cui si ottiene

$x = -A^{-1}bu$

Occorre prima definire le matrici del sistema

```
>> b=[1; 1;1];                  % Vettore b

>> c=[1 1 1];                   % Vettore c

>> d=0;                         % Vettore d

>> u=1;                         % Vettore degli ingressi

>> xeq=-inv(A)*b*u              % Calcolo dell'equilibrio:

xeq =

     0.7308
     0.1154
    -0.0769
```

L'uscita è invece data da

$y = cx + du$

da cui

```
>> yeq=c*xeq+d*u

yeq =

     0.7692
```

Per simulare il comportamento occorre definire (con il comando `ss()`) il modello dinamico. Successivamente, grazie al comando `lsim()`, si potrà osservare l'andamento delle variabili di interesse (nota: considerando un sistema a tempo continuo, le traiettorie saranno linee continue).

```
>> sistema=ss(A,b,c,d)          % sistema a tempo continuo

a =
          x1    x2    x3
x1        -1    -1     2
x2        0     -2    10
x3        0     -10   -2

b =
         u1
x1       1
x2       1
x3       1

c =
            x1   x2    x3
y1           1    1    1

d =
```

```
          u1
y1         0

Continuous-time model.
```

Definisco il vettore degli istanti di tempo della simulazione (NOTA: la durata deve essere maggiore del tempo di risposta T_R del sistema).

```
>> t=0:0.001:7;
```

Definisco l'ingresso *u* costante e pari a 1 (la dimensione di *u* deve essere pari alla dimensione del vettore dei tempi)

```
>> u=ones(size(t));
```

Simulo il sistema:

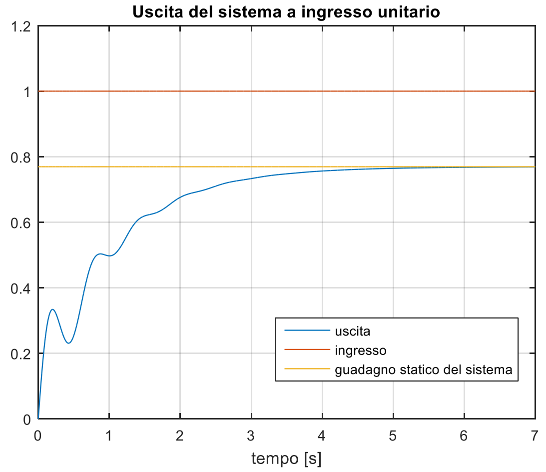
```
>> [uscita,tempo,stato]=lsim(sistema,u,t);
%NOTA: la condizione iniziale sullo stato è nulla; in alternativa va
specificata come X0 in lsim(sistema,u,t,X0)
```

Rappresento infine i risultati:

```
>> figure                      %apro una nuova figura
>> plot(tempo,uscita)          %uscita
>> hold all
>> plot(tempo,u)               %ingresso
>> ylim([0,1.2])               %visualizzo l'asse y da 0 a 1.2
>> grid on
>> title('Uscita del sistema a ingresso unitario')
>> xlabel('tempo [s]')
```

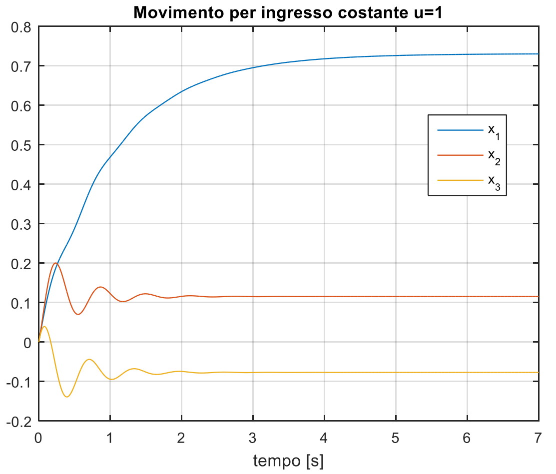
Rappresento anche il guadagno del sistema che è circa pari a 0.7692:

```
>> guadagno=dcgain(sistema)*ones(size(t));
>> plot(tempo,guadagno)
>> legend('uscita','ingresso','guadagno statico del sistema')
```



Rappresento ora il movimento:

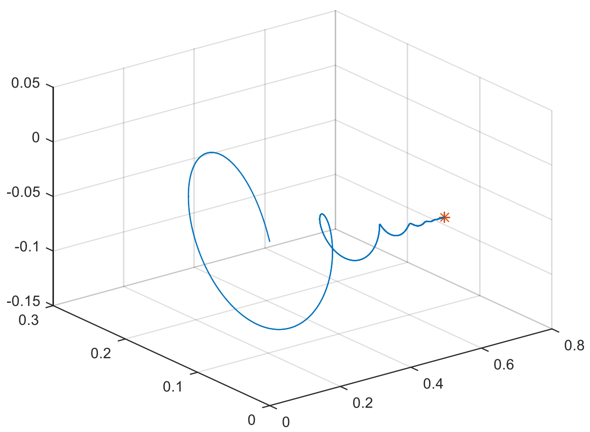
```
>> figure
>> plot(tempo,stato(:,1))  %prendo la prima
                           % colonna del vettore degli stati
>> hold all
>> plot(tempo,stato(:,2))
>> plot(tempo,stato(:,3))
>> grid on
>> title('Movimento per ingresso costante u=1')
>> xlabel('tempo [s]')
>> legend('x_1','x_2','x_3')
```



Infine, volendo tracciare le traiettorie nello spazio di stato, si utilizza il comando `plot3()` come segue

```
>> figure
>> plot3(stato(:,1),stato(:,2),stato(:,3))
```

I comandi `grid`, `hold on` e `plot3(xeq(1),xeq(2),xeq(3),'*')` permettono di sovrapporre lo stato di equilibrio che viene asintoticamente raggiunto.



Si definisca ora un nuovo ingresso come 5 volte l'ingresso precedente e si simuli il sistema in corrispondenza di questo nuovo ingresso:

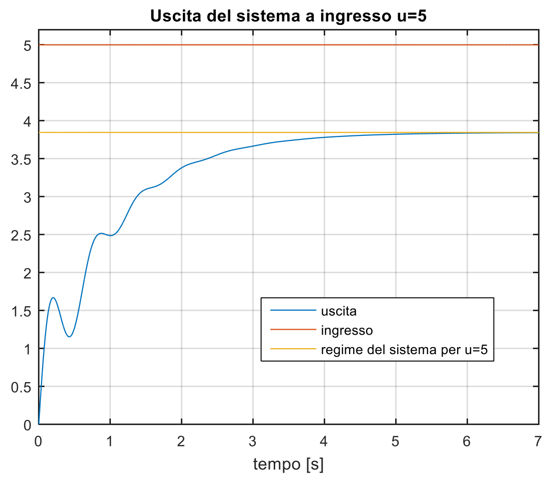
```
>> u2=5*u;
>> [uscita2,tempo2,stato2]=lsim(sistema,u2,t);
```

Rappresento i risultati:

```
>> figure                    %apro una nuova figura
>> plot(tempo2,uscita2)      %uscita
>> hold all
>> plot(tempo2,u2)           %ingresso
>> ylim([0,5.2])             %visualizzo l'asse y da 0 a 5.2
>> grid on
>> title('Uscita del sistema a ingresso u=5')
>> xlabel('tempo [s]')
```

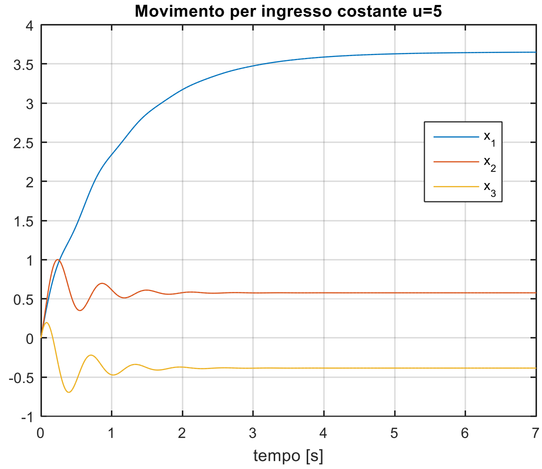
Rappresento anche l'uscita di regime: si noti che l'uscita di regime, essendo l'ingresso costante, non è altro che l'ingresso per il guadagno della funzione di trasferimento del sistema

```
>> regime=5*guadagno;
>> plot(tempo2,regime)
>> legend('uscita','ingresso','regime del sistema per u=5')
```



Rappresento ora il movimento:

```
>> figure
>> plot(tempo2,stato2(:,1))            %prendo la prima
                                       % colonna del vettore degli stati
>> hold all
>> plot(tempo2,stato2(:,2))
>> plot(tempo2,stato2(:,3))
>> grid on
>> title('Movimento per ingresso costante u=5')
>> xlabel('tempo [s]')
>> legend('x_1','x_2','x_3')
```



Come si può notare la risposta è del tutto analoga a quella trovata al punto precedente, fatto salvo per un fattore moltiplicativo 5; ciò era prevedibile, viste le proprietà di linearità del sistema.
Inoltre l'uscita di regime è pari al nuovo ingresso 5 moltiplicato per il guadagno del sistema.

Esercizio 2

La stabilità del sistema dipende dagli autovalori della matrice dinamica *A*: in particolare il sistema è *asintoticamente stabile* se e solo se tutti gli autovalori di *A* hanno modulo inferiore a 1.
Quindi, occorre innanzitutto definire la matrice *A*, per poi calcolarne gli autovalori tramite la funzione `eig()`. Successivamente si può calcolarne il modulo tramite la funzione `abs()`.
Pertanto, si può scrivere:

```
>> A=[-0.8 1 -1;                % matrice A
-0.5 0.6 -1;
-1 1 -0.5];

>> autoval=eig(A)               % autovalori di A

autoval =
           -0.4755 + 0.3641i
           -0.4755 - 0.3641i
           0.2509

>> modautoval=abs(autoval)      % modulo degli autovalori di A

modautoval =
     0.5989
     0.5989
     0.2509
```

Come si vede, il modulo dei tre autovalori è strettamente minore di 1, quindi il sistema è asintoticamente stabile. Inoltre, dato che il sistema ha due autovalori complessi coniugati, ci si possono aspettare oscillazioni smorzate.

Il tempo di risposta del sistema sarà pari a $T_R = 5\left(-{}^{1}/_{ln(|\lambda_D|)}\right)$ dove λ_D è l'autovalore dominate del sistema cioè quello/i di modulo massimo. Pertanto $T_R = 5\left(-{}^{1}/_{ln(|0.5989|)}\right)$ = 9.753 circa.

Per calcolare l'equilibrio del sistema sottoposto a ingresso costante pari a 2 occorre porre:

$x = Ax + bu$ e $y = cx + du$

da cui si ottiene $\bar{x} = (I - A)^{-1}b\bar{u}$ e $\bar{y} = c(I - A)^{-1}b\bar{u} + d\bar{u}$.

Per invertire la matrice $(I - A)$ (con il comando `inv()`) occorre innanzitutto costruire una matrice identità delle stesse dimensioni di *A*; per questo sono utili sia il commando `eye()` che il commando `size()`:

```
>> b=[1;                    % Vettore b
1;
1];
>> c=[1 1 1];               % Vettore c
>> d=0;                     % Vettore d
>> u=2;                     % Vettore degli ingressi
```

Occorre ora una matrice identità – `eye()`– delle dimensioni di A – `size()` –; pertanto mi occorre `eye(size(A))`; quindi l'equilibrio si può calcolare come:

```
>> xeq=inv(eye(size(A))-A)*b*u

xeq =
```

```
      0.8092
      0.7514
      1.2948


>> yeq=c*xeq+d*u

yeq =

      2.8555
```

Per simulare il comportamento del sistema occorre innanzitutto definire il modello dinamico a tempo discreto in `matlab` e poi utilizzare il comando `lsim()`.
Attenzione, essendo un sistema a tempo discreto, le traiettorie e il movimento saranno sequenze di punti, non linee continue.

I comandi che occorre usare sono `ss()` per la definizione del sistema dinamico e, appunto, `lsim()`; in quest'ultimo comando occorre specificare, oltre al sistema da simulare, l'ingresso al sistema (nel nostro caso $u = 2$) unitamente al vettore dei tempi che lo ha generato. Nel comando `ss()`, invece, bisogna specificare le matrici del sistema e, nel caso di sistema a tempo discreto, il tempo di campionamento, in questo esercizio unitario (il valore -1 indica un tempo di campionamento generico). Dunque si può scrivere:

```
>> sistema=ss(A,b,c,d,1)        %sistema a tempo discreto - tsampling =
1
a =
     x1  x2  x3
x1  -0.8  1  -1
x2  -0.5  0.6  -1
x3  -1  1  -0.5

b =
   u1
x1  1
x2  1
x3  1

c =
  x1  x2  x3
y1  1  1  1

d =
  u1
y1  0

Sampling time: 1

Discrete-time model.
```

Creo ora un ingresso costante che duri un certo numero di campioni (superiore al tempo di risposta del sistema)

```
>> t=0:20;                    %vettore di tempi a passo unitario
>> u=2*ones(size(t));         %ingresso: vale sempre due
```

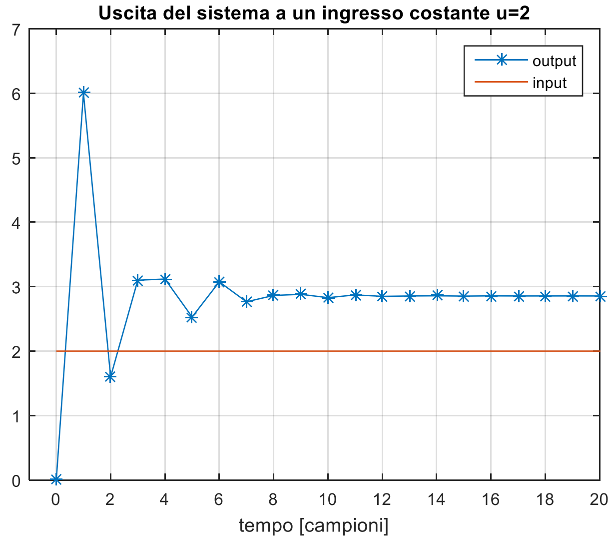
Simulo il sistema, salvando uscita, tempo e traiettorie degli stati:

```
>> [out,time,state]=lsim(sistema,u,t);
```

NOTA: la condizione iniziale sullo stato è nulla; in alternativa va specificata come X0 in `lsim(sistema,u,t,X0)`.

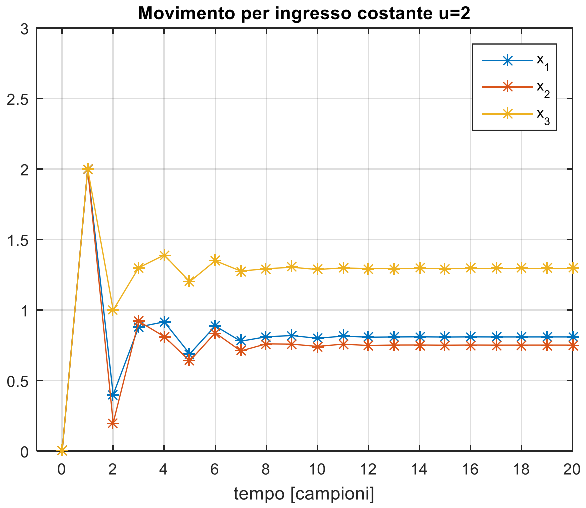
Rappresento i risultati:

```
>> figure                    %apro una nuova figura
>> plot(time,out,'-*')       %uscita
>> hold all
>> plot(time,u)              %ingresso
>> xlim([-1,20])             %visualizzo l'asse x da -1 a 20
>> ylim([0,7])               %visualizzo l'asse y da 0 a 7
>> grid on
>> title('Uscita del sistema a un ingresso costante u=2')
>> xlabel('tempo [campioni]')
>> legend('output','input')       %aggiungo la legenda
```



Rappresento l'andamento dei tre stati:

```
>> figure
>> plot(time,state(:,1),'-*')          %prendo la prima colonna
                                       %del vettore degli stati
>> hold all
>> plot(time,state(:,2),'-*')
>> plot(time,state(:,3),'-*')
>> xlim([-1,20])
>> ylim([0,3])
>> grid on
>> title('Movimento per ingresso costante u=2')
>> xlabel('tempo [campioni]')
>> legend('x_1','x_2','x_3')
```



Infine, volendo tracciare le traiettorie nello spazio di stato (sequenza di punti), si utilizza il comando `plot3()` come segue

```
>> figure
>> grid
>> plot3(state(:,1),state(:,2),state(:,3),'*')
```

Esercizio 3

Innanzitutto occorre scrivere il sistema dinamico nella forma:

$$\dot{x}(t) = Ax(t) + bu(t)$$

$$y(t) = cx(t) + du(t)$$

Per scrivere il sistema dinamico in forma matriciale si può porre

$$x_1(t) = x(t)$$

$$x_2(t) = \dot{x}(t)$$

A questo punto si può scrivere:

$$\dot{x}_1(t) = x_2(t)$$
$$\dot{x}_2(t) = \frac{1}{M_S}\left(-a_S x_2(t) - k_S x_1(t)\right)$$
$$y(t) = x_1(t)$$

da cui

$A = \begin{vmatrix} 0 & 1 \\ -\frac{k_S}{M_S} & -\frac{a_S}{M_S} \end{vmatrix}$ $\qquad b = \begin{vmatrix} 0 \\ 0 \end{vmatrix}$ $\qquad c = \begin{vmatrix} 1 & 0 \end{vmatrix}$ $\qquad d = 0$

Per studiare la stabilità del sistema, occorre conoscere gli autovalori della matrice *A*.
Il polinomio caratteristico del sistema è:

$$det(\lambda I - A) = det\begin{vmatrix} \lambda & -1 \\ \frac{k_S}{M_S} & \lambda + \frac{a_S}{M_S} \end{vmatrix} = \lambda^2 + \frac{a_S}{M_S}\lambda + \frac{k_S}{M_S}$$

da cui si possono calcolare gli autovalori: $\lambda_{1,2} = \frac{-a_S \pm \sqrt{a_S^2 - k_S^2}}{2M_S}$.

Essendo sia a_S che k_S positivi, gli autovalori avranno sempre parte reale negativa. Pertanto il sistema è sempre asintoticamente stabile (come doveva essere rappresentando un sistema dissipativo).
Controlliamo numericamente che la soluzione analitica sia corretta.
Assegniamo i valori dati alle variabili del problema

```
>> M_s = 250;                    % Kg
>> k_s = 5000;                   % N/m
>> a_s = 1000;                   % N*s/m
```

Ora scriviamo la matrice *A* e valutiamone gli autovalori:

```
>> A=[0 1;
-k_s/M_s -a_s/M_s];

>> autoval=eig(A)

     -2.0000 + 4.0000i
     -2.0000 - 4.0000i
```

Quindi, essendo entrambi gli autovalori a parte reale negativa, il sistema risulta asintoticamente stabile.

Per risolvere il secondo punto occorre riscrivere l'equazione del sistema.

Nell’equazione precedente, infatti, la variabile *x* descriveva la dinamica della massa (sedile + passeggero) quando l’auto era ferma. Per studiare i casi in cui l’auto si muove verticalmente a causa del manto stradale irregolare occorre riscrivere l'equazione tenendo conto che il moto del sedile sia relativo al moto del manto stradale, ovvero scrivendo:

$$M_S \ddot{x}_{sedile} = -k_S x_{rel} - a_S \dot{x}_{rel}$$

dove:

$$x_{rel} = x_{sedile} - x_{strada}$$

Quindi, l'equazione dinamica diventa:

$$M_S \ddot{x}_{sedile} + a_S \dot{x}_{sedile} + k_S x_{sedile} = k_S x_{strada} + a_S \dot{x}_{strada}$$

Considerando come ingresso la forma della strada si ottiene un sistema dinamico della forma

$$\dot{x}_1(t) = x_2(t)$$
$$\dot{x}_2(t) = \frac{1}{M_S}\left(-a_S x_2(t) - k_S x_1(t)\right) + \frac{a_S \dot{x}_{strada} + k_S x_{strada}}{M_S}$$
$$y(t) = x_1(t)$$

dunque si scriverà l'ingresso come:

$$u = \begin{vmatrix} x_{strada} \\ \dot{x}_{strada} \end{vmatrix}$$

B sarà dunque una matrice della forma:

$$B = \begin{vmatrix} 0 & 0 \\ \frac{k_S}{M_S} & \frac{a_S}{M_S} \end{vmatrix}$$

Si può ora scrivere il nuovo sistema, aggiornando i valori delle matrici *B* e *d*:

```
>> B=[0 0;
k_s/M_s a_s/M_s];
>> c=[1 0];
>> d=[0 0];

sys_sedile_2=ss(A,B,c,d)
A =
  x1 x2
x1 0 1
x2 -20 -4

B =
  u1 u2
x1 0 0
x2 20 4

c =
  x1 x2
y1 1 0

d =
  u1 u2
y1 0 0

Continuous-time model.
```

A questo punto, per simulare il sistema, occorre costruire correttamente l'ingresso e la sua derivata. La funzione `genera_rampa(vel)` (digita `help genera_rampa`) modellizza una rampa a pendenza costante e di lunghezza (orizzontale) pari a 1 m e altezza 10 cm (*hdosso*). Chiamando *d* la lunghezza della rampa si può modellizzare la rampa (cioè l'altezza del manto stradale) in funzione della velocità di percorrenza del veicolo (v_{car}):

$$x_{strada}(t) = \begin{cases} 0 & t < t_0 \\ \dfrac{h_{dosso}}{d_{dosso}} v_{car}(t - t_0) & t_0 < t < t_1 \\ h_{dosso} & t > t_1 \end{cases}$$

dove il tempo di salita, dopo aver fissato arbitrariamente t_0, è così calcolato:

$$t_1 = t_0 + \frac{d_{dosso}}{v_{car}}$$

Inoltre

$$\dot{x}_{strada}(t) = \begin{cases} 0 & t < t_0 \\ \dfrac{h_{dosso}}{d_{dosso}} v_{car} & t_0 < t < t_1 \\ 0 & t > t_1 \end{cases}$$

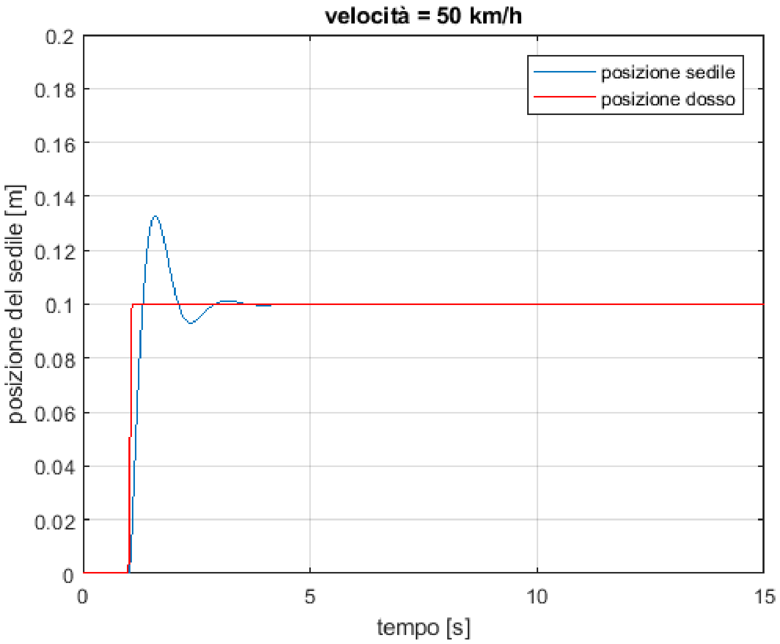
Dunque avendo tutti i dati è possibile generare il vettore di ingresso $u = \begin{vmatrix} x_{strada} \\ \dot{x}_{strada} \end{vmatrix}$.

NOTA: le unità di misura da utilizzare affinché i conti siano corretti sono quelle del sistema internazionale (metri, secondi, m/s, etc...).

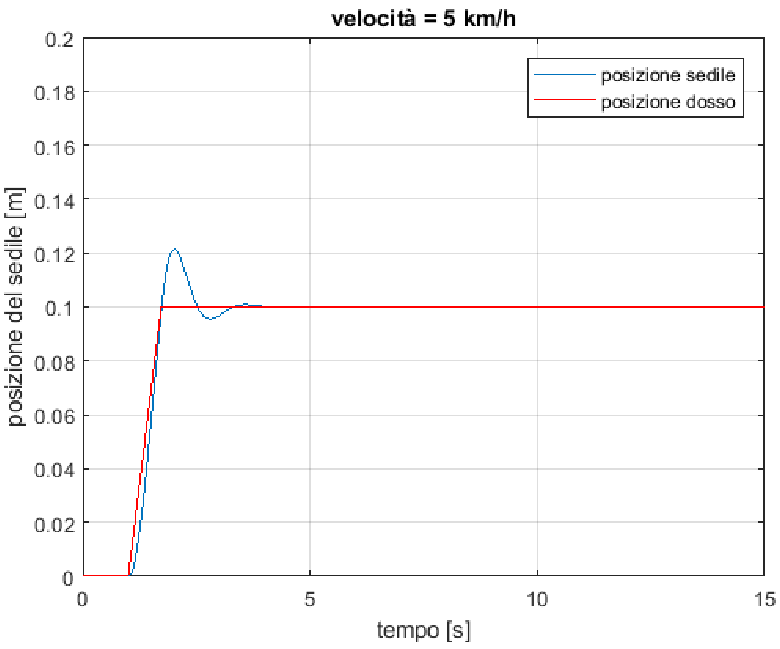
Si può pertanto risolvere l'ultimo punto nel modo seguente:

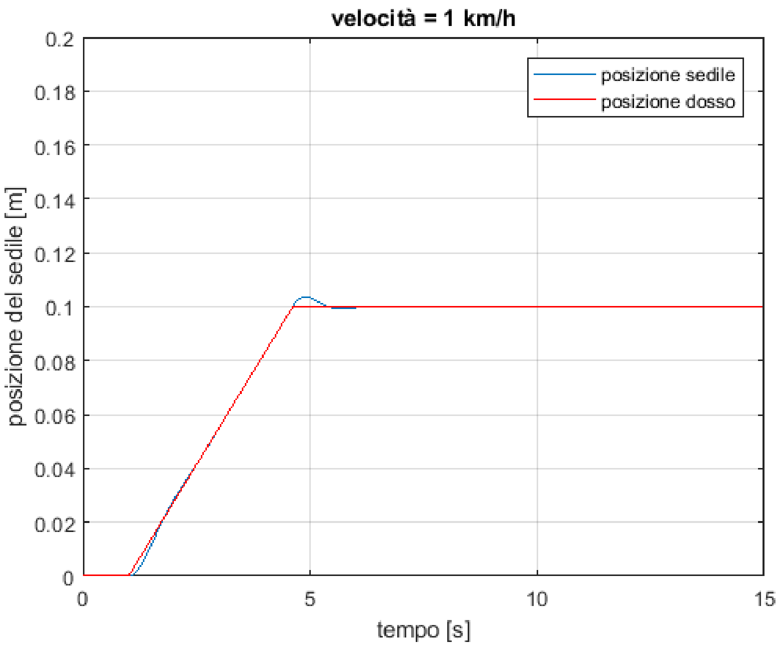
```
>> vel=50/3.6;                          % m/s
>> [x_strada,x_dot_strada,t_strada]=genera_dosso(vel);
>>[pos_sedile_2,t_sim,st_sim]=lsim(sys_sedile_2,[x_strada;x_dot_strada],t_strada);
```

da cui, tracciando solo gli andamenti del profilo stradale, cioè l'altezza del dosso (`x_strada`) e della posizione del sedile (`pos_sedile_2`), si ottiene:



Per simulare cosa succede ad altre velocità basta cambiare il valore di `vel`.


velocità = 5 km/h
posizione sedile
posizione dosso
posizione del sedile [m]
tempo [s]
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
5
10
15


velocità = 1 km/h
posizione sedile
posizione dosso
posizione del sedile [m]
tempo [s]
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
5
10
15