Progetto 0

Table of Contents


```
clear all;
clc;
close all;

system=ElasticRoboticSystem();

st=system.getSamplingPeriod;

cs=ControlledSystemScara(system,'Alfa');
```

1. Joint 2 Identification

Definisco la portante come un segnale sinusoidale a bassa frequenza, con un'ampiezza tale da non sforare i limiti. L'ideale è andare in prossimità dei limiti mantenendo un discreto margine (esempio 70% limiti).

```
omega_portante=0.5;
ampiezza_portante=20;
T_portante=2*pi/omega_portante;
```

scelgo una lunghezza dell'esperimento di almeno un periodo della portante.

```
t=(0:st:(1.1*T_portante))';
portante=ampiezza_portante*sin(omega_portante*t);
```

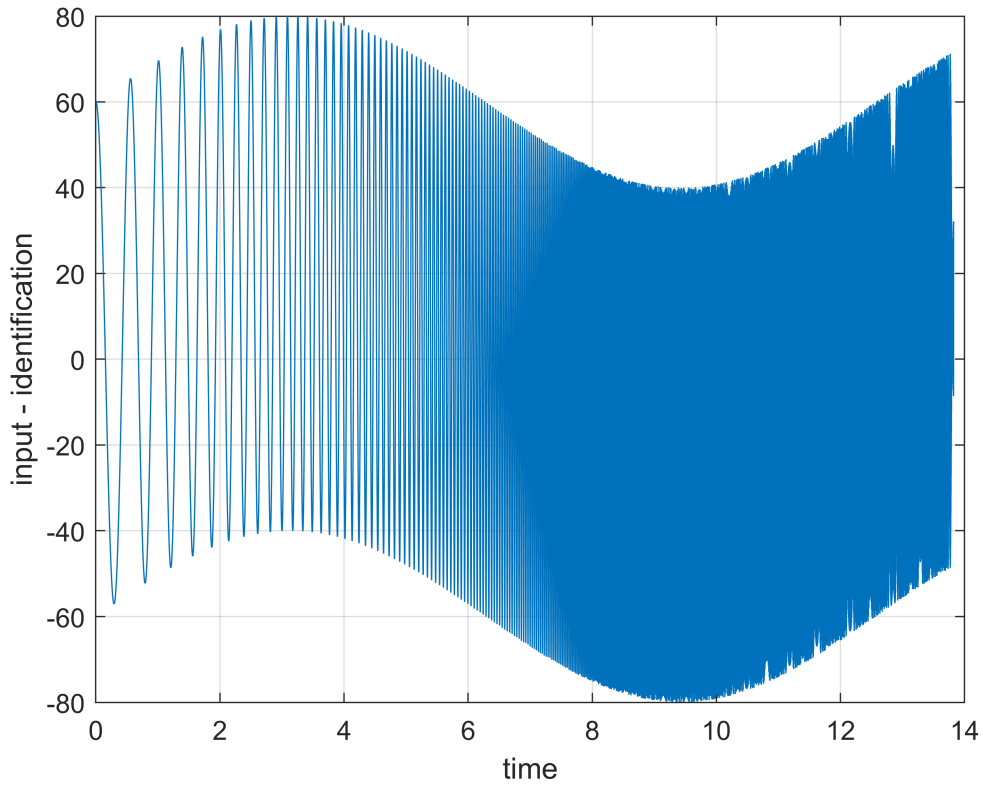
Dopo di che posso andare a definire il segnale eccitante, che partirà da una frequenza maggiore (almeno un decade).

```
w0=10; %rad/s
w1=pi/st; %rad/s
control_action_identificazione = chirp(t,w0/2/pi,t(end),w1/2/pi, 'logarithmic');
ampliezza_identificazione=60;
```

Sommo i due segnali

```
control_action=portante + control_action_identificazione*ampliezza_identificazione;
figure
plot(t,control_action)
xlabel('time')
```

```
ylabel('input - identification')
grid on
```



1.1 Simulazione sistema

I tratti con velocità vicina allo 0 hanno una dinamica chiaramente non lineare, l'approssimazione lineare nelle frequenze esplorare dal segnale eccitante in quei punti sarà scarsa.

```
system.show
```

```
This system has 4 outputs:
- position_1
- position_2
- velocity_1
- velocity_2
This system has 2 inputs:
- torque_1, with maximum limit = 600.000000
- torque_2, with maximum limit = 600.000000
ans =
  ElasticRoboticSystem with no properties.
```

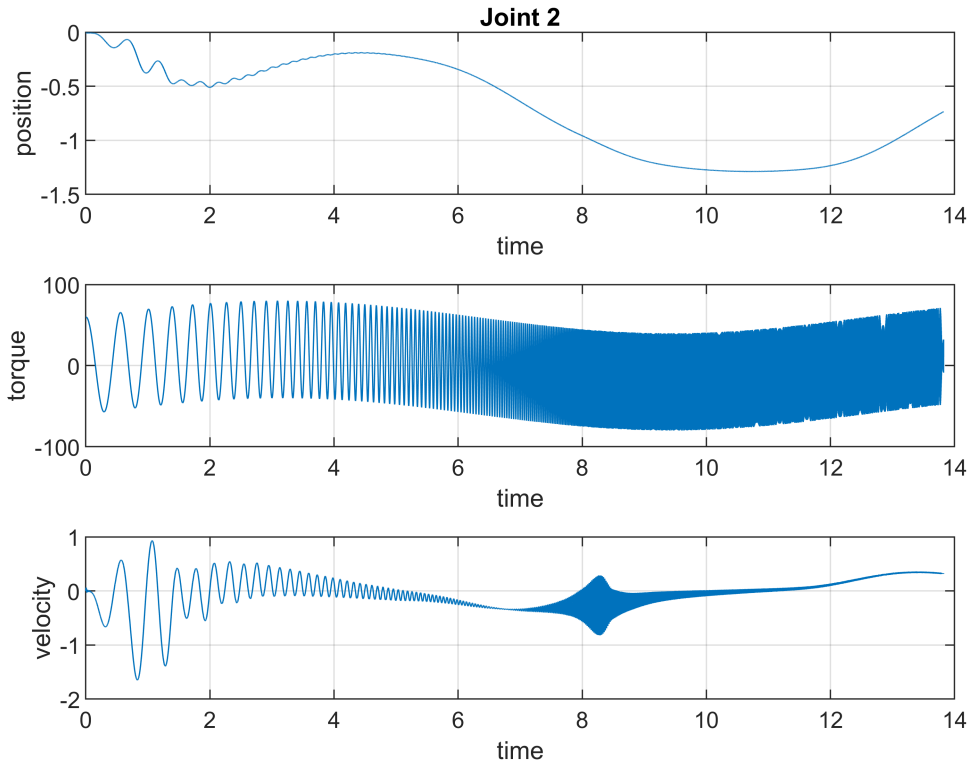
```
cs.initialize
tic
ref_j0=0;
x0=zeros(1,4);
k=50/(1/360*2*pi);
x=x0;
s=tf('s');
Ti=10;
PI=k*(1+1/(Ti*s));
```

```
pi_d=c2d(PI,st);
controller1=Controller(st,pi_d,[-600 600]);
torque=zeros(length(control_action),2);
process_output_id2=[];
for idx=1:length(t)
    still_ca = controller1.computeControlAction(ref_j0,x(1));
    [x,t(idx,1)]=cs.openloop([still_ca control_action(idx)]);
    process_output_id2(idx,:) = x;
    torque(idx,:) = [still_ca control_action(idx)];
end
toc
```

```
Elapsed time is 13.520654 seconds.
```

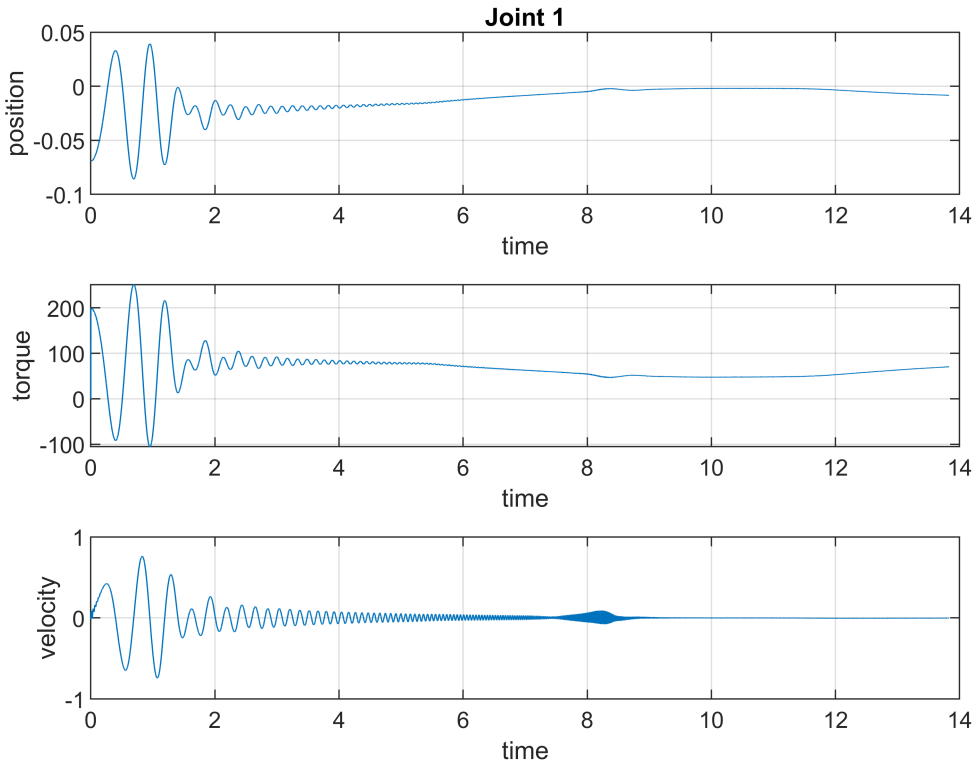
Joint 2

```
figure(1)
subplot(3,1,1)
plot(t,process_output_id2(:,2))
xlabel('time')
ylabel('position')
title('Joint 2')
grid on

subplot(3,1,2)
plot(t,torque(:,2))
xlabel('time')
ylabel('torque')
grid on

subplot(3,1,3)
plot(t,process_output_id2(:,4))
xlabel('time')
ylabel('velocity')
grid on
```



Joint 1

```
figure(2)
subplot(3,1,1)
plot(t,process_output_id2(:,1))
xlabel('time')
ylabel('position')
title('Joint 1')
grid on

subplot(3,1,2)
plot(t,torque(:,1))
xlabel('time')
ylabel('torque')
grid on

subplot(3,1,3)
plot(t,process_output_id2(:,3))
xlabel('time')
ylabel('velocity')
grid on
```



Nota: le frequenze in prossimità dei cambi di velocità sono stimate peggio (è opportuno che: 1) ripetere con segnali diversi 2) in validazione il signale sia diverso).

1.2 Calcolo risposta in frequenza

Definisco l'oggetto identification data (iddata)

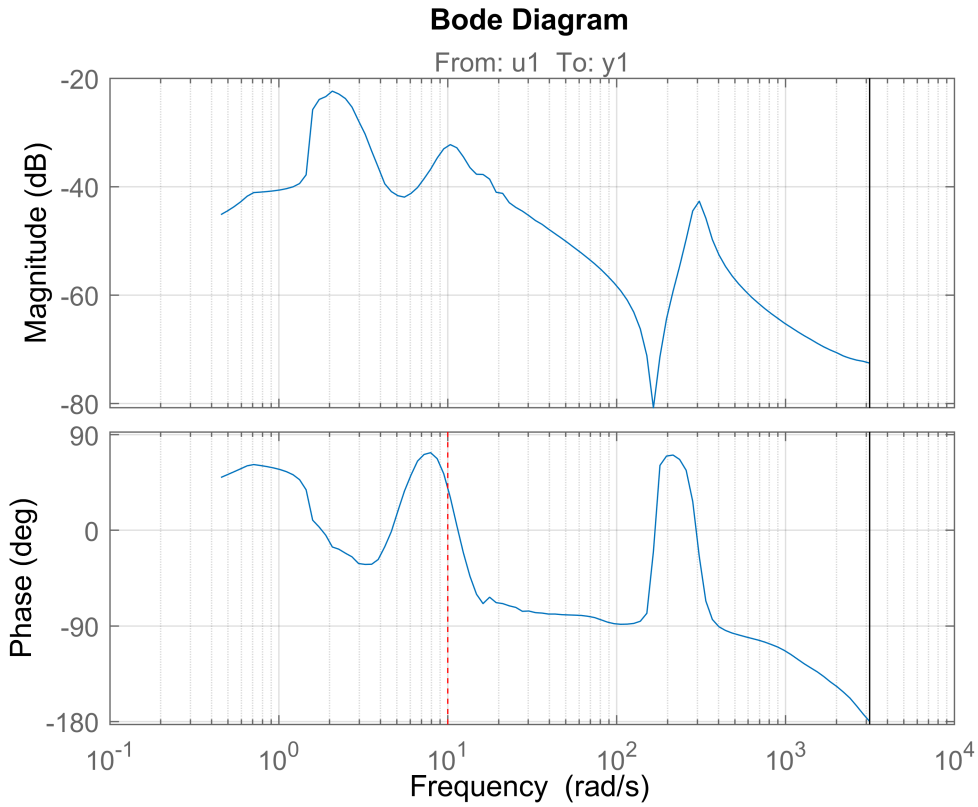
```
identification=iddata(process_output_id2(:,4),control_action,st);
```

Calcolo la risposta in frequenza con il comando e la plotto

```
freq_resp_ident = spafdr(identification);
figure(3)
bode_opts = bodeoptions('cstprefs');
bode_opts.PhaseWrapping = 'on';
h=bodeplot(freq_resp_ident, bode_opts);
grid on
hold on
```

Plotto la frequenza di inizio del segnale eccitante

```
hold on
plot(w0*[1 1],ylim,'--r')
hold off
grid on
```



La parte a sinistra nella linea tratteggiate non mi interessa (il segnale non ha contribuito informatico apprezzabile ed è "disturbato" dalla presenza dell'attrito statico).

1.3 Stima del modello

devo dire al modello di non considerare le basse frequenze. Creo un vettore di pesi da dare allo stimatore (peso basso, poca importanza).

```
peso=ones(length(freq_resp_ident.Frequency),1);
wpeso0=50;
wpeso1=1000;
peso(freq_resp_ident.Frequency<wpeso0)=1e-5;
peso(freq_resp_ident.Frequency>wpeso1)=1e-5;

opts=ssestOptions('WeightingFilter',peso,'EnforceStability',1);
modello_continuo = ssest(freq_resp_ident,3,opts);
modello_discreto = ssest(freq_resp_ident,3,'Ts',st,opts);
modello_continuo_tf=tf(modello_continuo)
```

```
modello_continuo_tf =

  From input "u1" to output "y1":
       0.5528 s^2 + 4.62 s + 1.52e04
  ---------------------------------------
  s^3 + 61.68 s^2 + 9.057e04 s + 1.101e06

Continuous-time transfer function.
```

```
save modello modello_continuo_tf
```

1.4 Grafico modelli

aggiungo opzione per evitare la molteplicità di 360° nella base

```
figure
bode_opts = bodeoptions('cstprefs');
bode_opts.PhaseWrapping = 'on';
```

plotto i dati sperimentali

```
h=bodeplot(freq_resp_ident, bode_opts);
```

si può aggiungere l'intervallo di confidenza (in questo caso 3*sigma)

```
hold on
showConfidence(h,3)
```

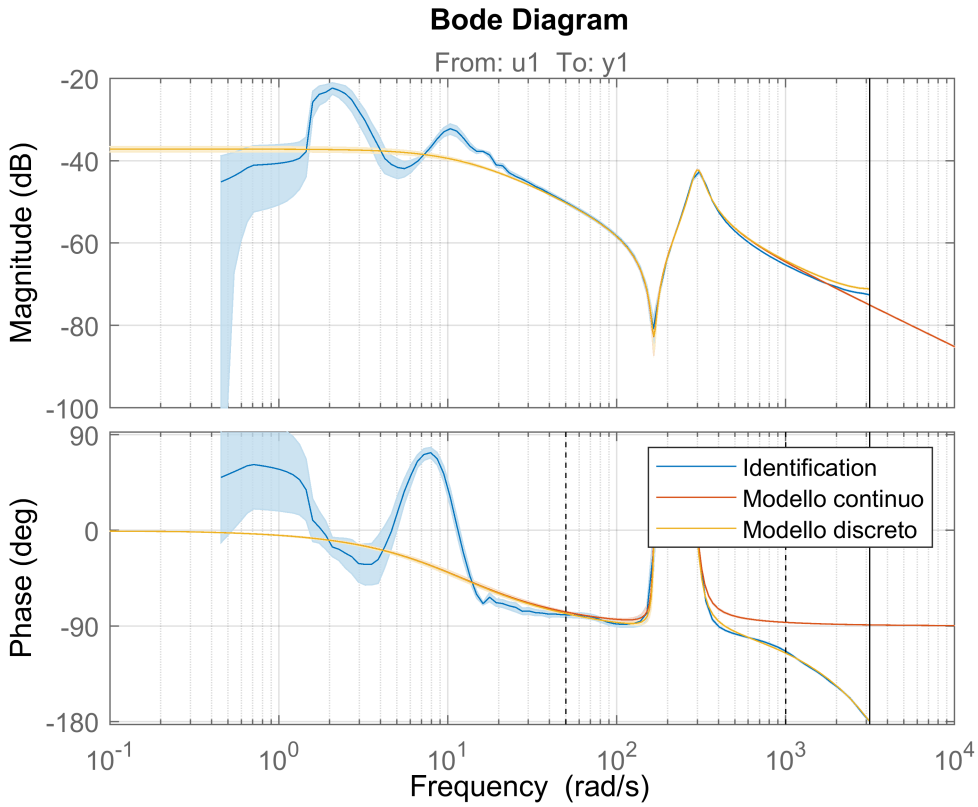
grafico i modelli stimati

```
bode(modello_continuo,modello_discreto, bode_opts)
```

visualizzo solo il range di frequenze di interesse

```
xlim([w0 w1])
plot([wpeso0 wpeso0],ylim,'--k')
plot([wpeso1 wpeso1],ylim,'--k')
```

```
hold off
grid on
xlim([1e-1 1e4])

legend('Identification','Modello continuo','Modello discreto')
```



2. Joint 1 Identification

2.1 Controller joint 2

```
figure
G = tf(modello_continuo);
```

Vorrei un ω_c prossima all'antirisonanza (che è attorno a 100 rad/s)

```
wc=80;
```

L'azione integrale mi sfasa, la metto almeno una decade prima.

```
wi=wc/20; % pulsazione azione integrale
Ti=1/wi;
```

Il mio controllore sarà

$C(s) = K_p(1 + 1/(Ti \cdot s)) = K_p C_o(s)$

con $C_o(s) =$

```
s=tf('s');
Co=1+1/(Ti*s);
```

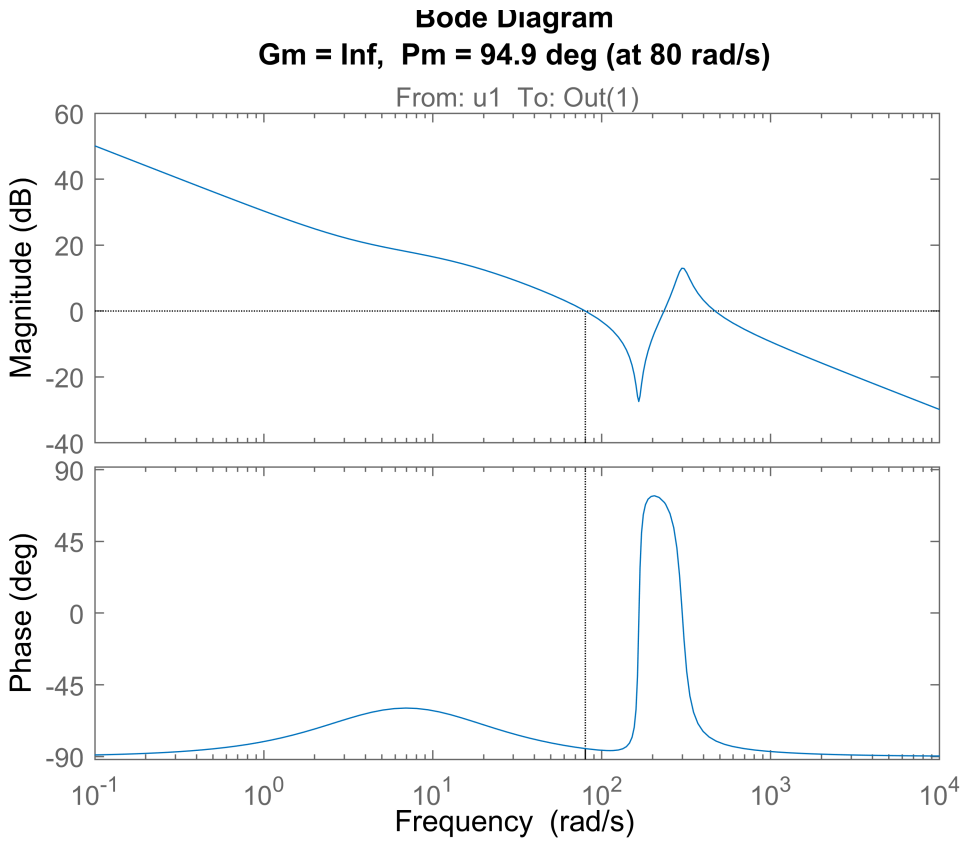
Calcolo K_p in modo che $C(s) * P(s) = K_p(C_o(s) * P(s))$ abbia modulo 1 in ω_c.

$|K_p(C_o(j\omega_c) * P(j\omega_c))| = 1$ quindi $K_p = 1/|(C_o(j\omega_c) * P(j\omega_c))|$

```
Kp=1/abs(freqresp(Co*G,wc));
C=Kp*Co
```

```
C =

  144.8 s + 579.4
  ---------------
      0.25 s

Continuous-time transfer function.
```

```
figure
margin(C*G)
```



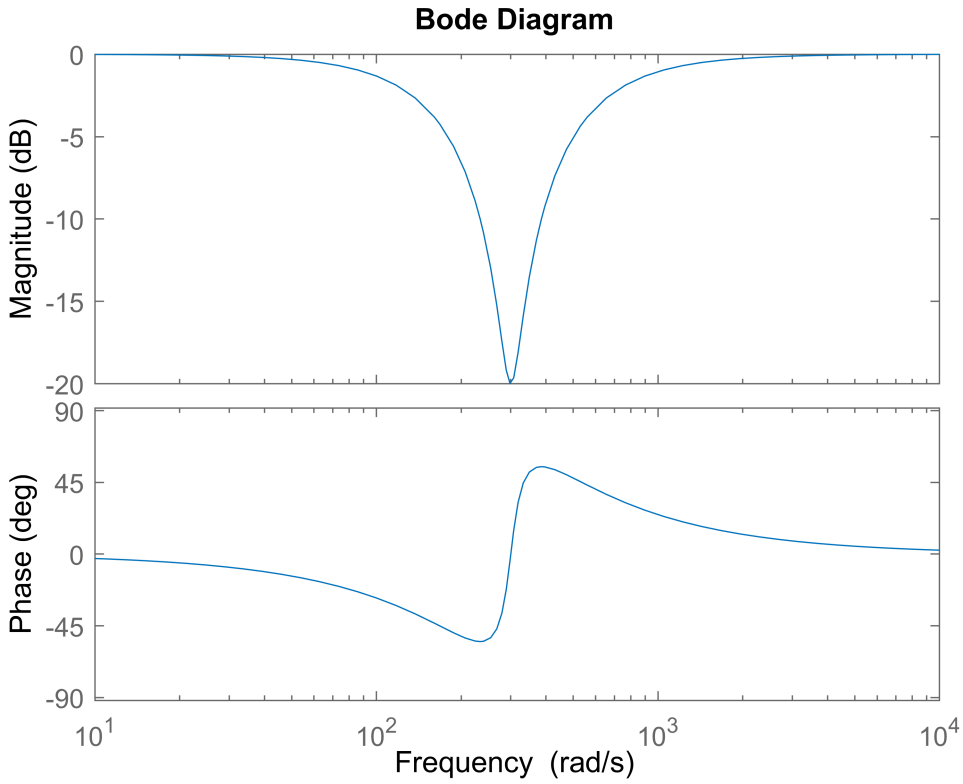
2.2 Filtro notch

```
damp(G)
```

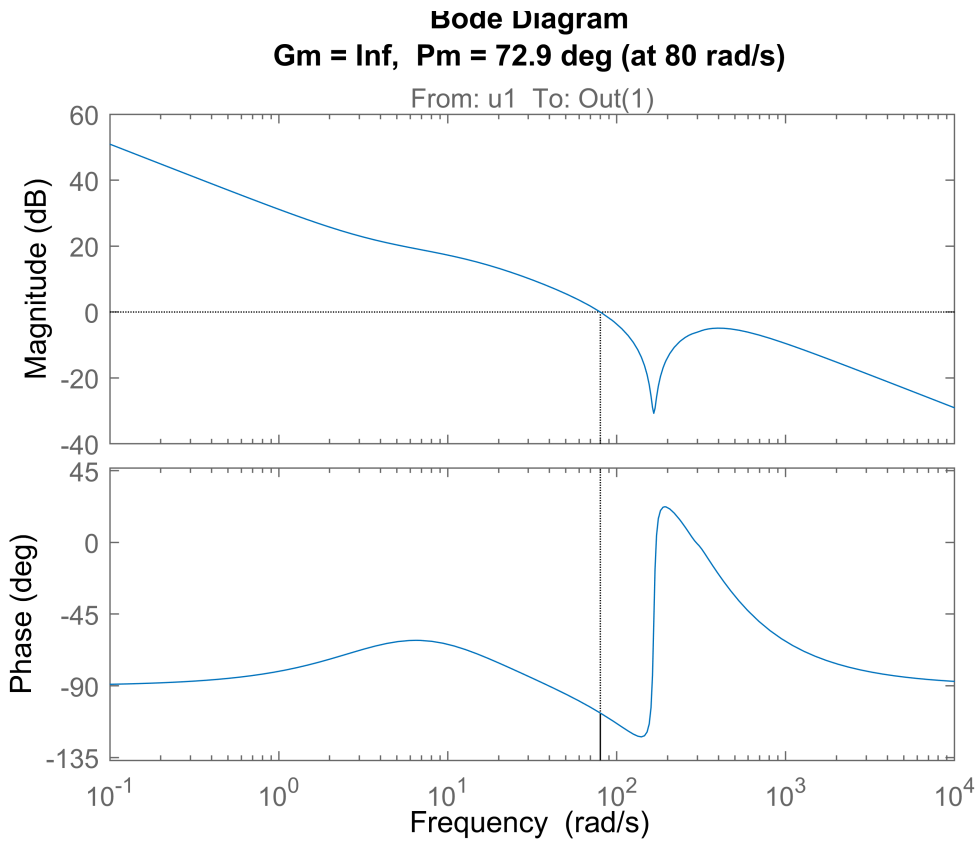
```
        Pole              Damping       Frequency      Time Constant
                                      (rad/seconds)       (seconds)

 -1.22e+01                1.00e+00       1.22e+01         8.17e-02
 -2.47e+01 + 2.99e+02i    8.24e-02       3.00e+02         4.05e-02
 -2.47e+01 - 2.99e+02i    8.24e-02       3.00e+02         4.05e-02
```

```
wn=300;
```

```
xci_z=0.08; %aggiustate xci_z e xci_p per rendere il filtro più o meno selettivo
xci_p=0.8;
notch=(s^2+2*xci_z*wn*s+wn^2)/(s^2+2*xci_p*wn*s+wn^2);
figure
bode(notch)
```




```
Kp=1/abs(freqresp(Co*G*notch,wc));
C=Kp*Co*notch;
figure
margin(C*G)
```




```
getPeakGain(feedback(1,C*G))
```

```
ans = 1.1407
```

2.3 Generazione Chirp

Costruzione segnale Chirp per primo giunto

```
omega_portante_j1=1;
ampiezza_portante_j1=0.5;
T_portante=2*pi/omega_portante_j1;
```

scelgo una lunghezza dell'esperimento di almeno un periodo della portante.

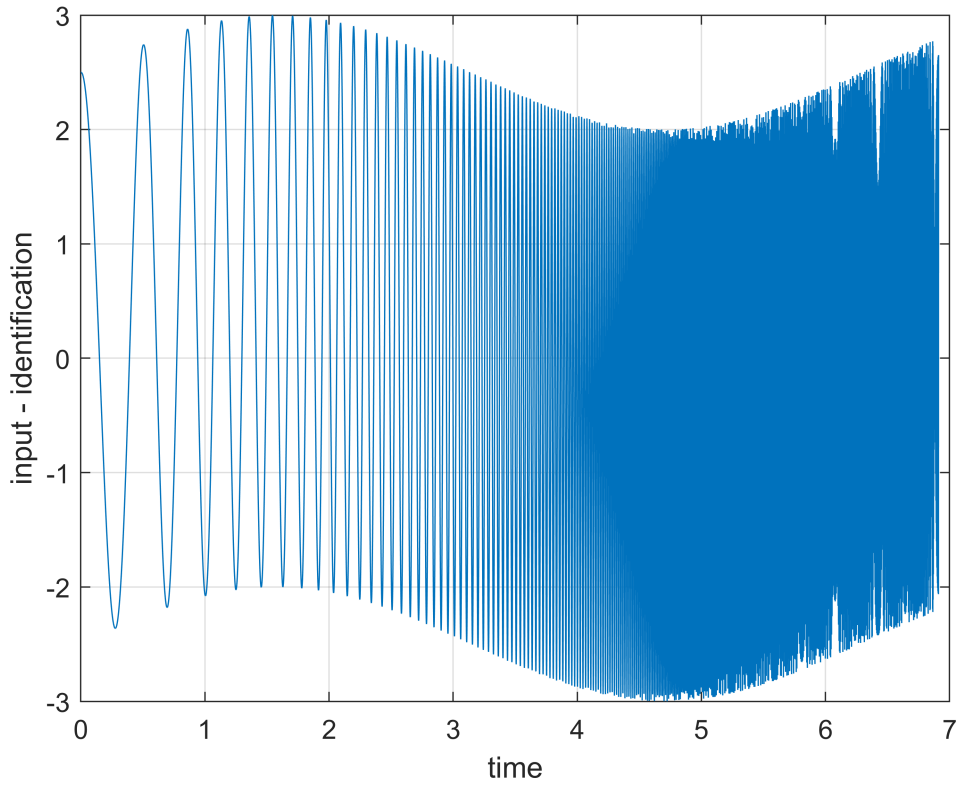
```
t=(0:st:(1.1*T_portante))';
portante=ampiezza_portante_j1*sin(omega_portante_j1*t);
```

Dopo di che posso andare a definire il segnale eccitante, che partirà da una frequenza maggiore (almeno un decade).

```
w0=10; %rad/s
w1=pi/st; %rad/s
control_action_identificazione_j1 = chirp(t,w0/2/pi,t(end),w1/2/pi, 'logarithmic');
ampliezza_identificazione_j1=2.5;
```

Sommo i due segnali

```
control_action_j1=portante + control_action_identificazione_j1*ampliezza_identificazione_j1;
figure
plot(t,control_action_j1)
xlabel('time')
ylabel('input - identification')
grid on
```



Identificazione giunto 1

```
system.show
```

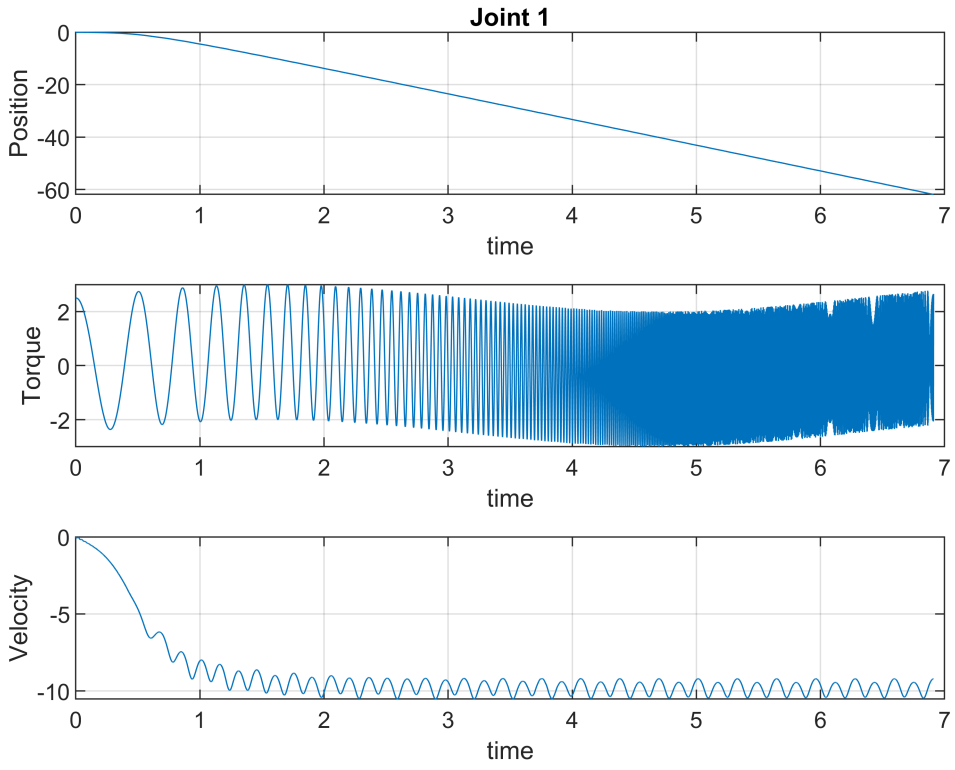
```
This system has 4 outputs:
- position_1
- position_2
- velocity_1
- velocity_2
This system has 2 inputs:
- torque_1, with maximum limit = 600.000000
- torque_2, with maximum limit = 600.000000
ans =
  ElasticRoboticSystem with no properties.
```

```
cs.initialize
tic
ref_j0=0;
x0=zeros(1,4);
x=x0;

pi_d=c2d(C,st);
```

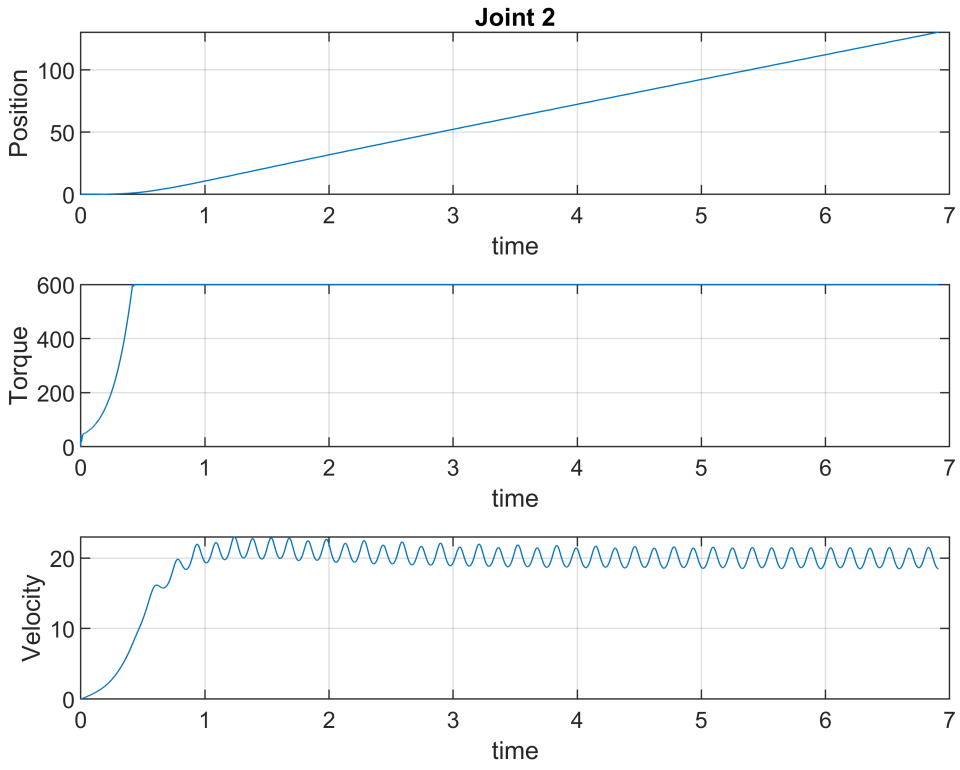
```
controller1=Controller(st,pi_d,[-600 600]);
torque=zeros(length(control_action_j1),2);
process_output_id1=[];
for idx=1:length(t)
    still_ca = controller1.computeControlAction(ref_j0,x(1));
    [x,t(idx,1)]=cs.openloop([control_action_j1(idx) still_ca]);
    process_output_id1(idx,:) = x;
    torque(idx,:) = [control_action_j1(idx) still_ca];
end
toc
```

```
Elapsed time is 7.136612 seconds.
```

```
figure
subplot(3,1,1)
plot(t,process_output_id1(:,1))
xlabel('time')
ylabel('Position')
title('Joint 1')
grid on

subplot(3,1,2)
plot(t,torque(:,1))
xlabel('time')
ylabel('Torque')
grid on

subplot(3,1,3)
plot(t,process_output_id1(:,3))
xlabel('time')
ylabel('Velocity')
grid on
```




```
figure
subplot(3,1,1)
plot(t,process_output_id1(:,2))
xlabel('time')
ylabel('Position')
title('Joint 2')
grid on

subplot(3,1,2)
plot(t,torque(:,2))
xlabel('time')
ylabel('Torque')
grid on

subplot(3,1,3)
plot(t,process_output_id1(:,4))
xlabel('time')
ylabel('Velocity')
grid on
```


Joint 2
Position
100
50
0
0
1
2
3
4
5
6
7
time
Torque
600
400
200
0
0
1
2
3
4
5
6
7
time
Velocity
20
10
0
0
1
2
3
4
5
6
7
time