POLITECNICO DI MILANO





Laboratorio di Fondamenti di Automatica

Ingegneria Elettrica

Sessione 2/3

Danilo Caporale [caporale@elet.polimi.it]

Outline

- Funzione di trasferimento e risposta in frequenza
- Diagrammi di Bode e teorema della risposta in frequenza
- Diagramma polare e di Nyquist
- Stabilità: criterio di Bode e di Nyquist
- Sistema con ritardo di tempo: un esempio
- Proprietà bloccante degli zeri

- Nota: ci servirà l'editor di testo. Apritelo col comando edit.

Funzione di trasferimento e risposta in frequenza

Definizioni (richiamo)

Dato il sistema in forma di stato:

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

si definisce **funzione di trasferimento** la funzione di variabile complessa s:

$$G(s) = C\left(sI - A\right)^{-1} B + D$$

si definisce **risposta in frequenza** associata al sistema:

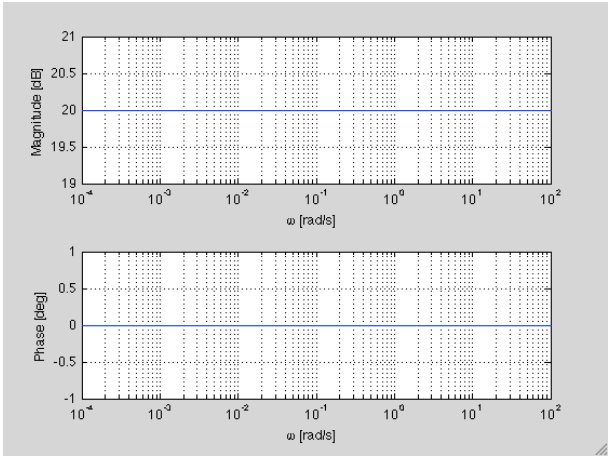
$$G(j\omega) = C\left(j\omega I - A\right)^{-1} B + D$$

La risposta in frequenza coincide* con la funzione di trasferimento calcolata in: $s = j\omega, \quad \omega > 0$

* salvo poli sull'asse immaginario

Diagrammi di Bode

Calcoliamo i diagrammi di modulo e fase della risposta in frequenza con Matlab, in alcuni semplici casi:



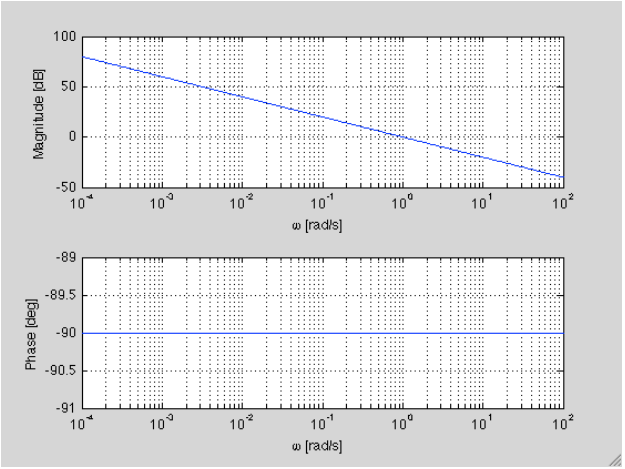
Guadagno puro: $G(s) = 10$

```
>> num = 10;                    % numeratore
>> den = 1;                     % denominatore
>> j = sqrt(-1);                % unità immaginaria
>> w = logspace(-4, 2, 500);    % asse delle ascisse (pulsazioni)
>> R = polyval(num,j*w)./polyval(den,j*w); % calcolo effettivo della risposta in frequenza
>> modR = abs(R);               % modulo della risposta in frequenza
>> faseR = angle(R);            % fase della risposta in frequenza
>> subplot(2,1,1); semilogx(w,20*log10(modR)); grid on;
>> subplot(2,1,2); semilogx(w,faseR*180/pi); grid on;
```

Diagrammi di Bode

Calcoliamo i diagrammi di modulo e fase con Matlab, in alcuni semplici casi:

Integratore: $$G(s) = \frac{1}{s}$$

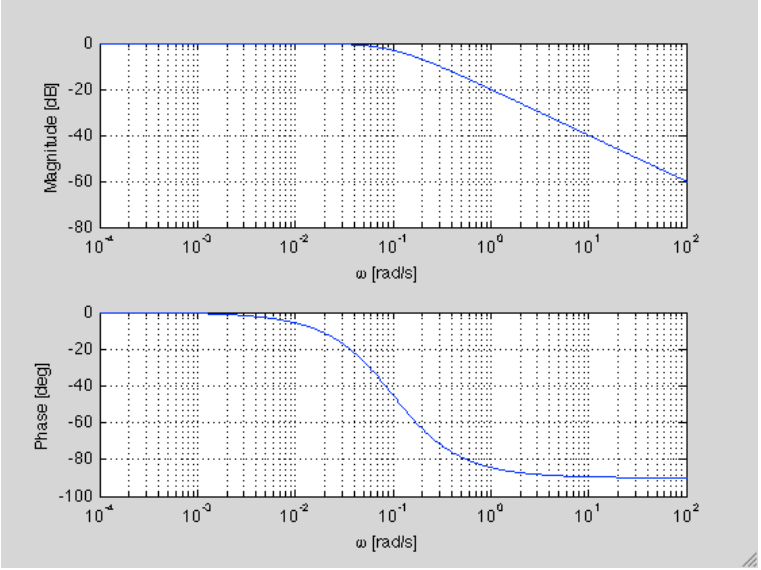



```
>> num = 1;
>> den = [1 0];
>> j = sqrt(-1);
>> w = logspace(-4, 2, 500);
>> R = polyval(num,j*w)./polyval(den,j*w);
>> modR = abs(R);
>> faseR = angle(R);
>> subplot(2,1,1); semilogx(w,20*log10(modR)); grid on;
>> subplot(2,1,2); semilogx(w,faseR*180/pi); grid on;
```

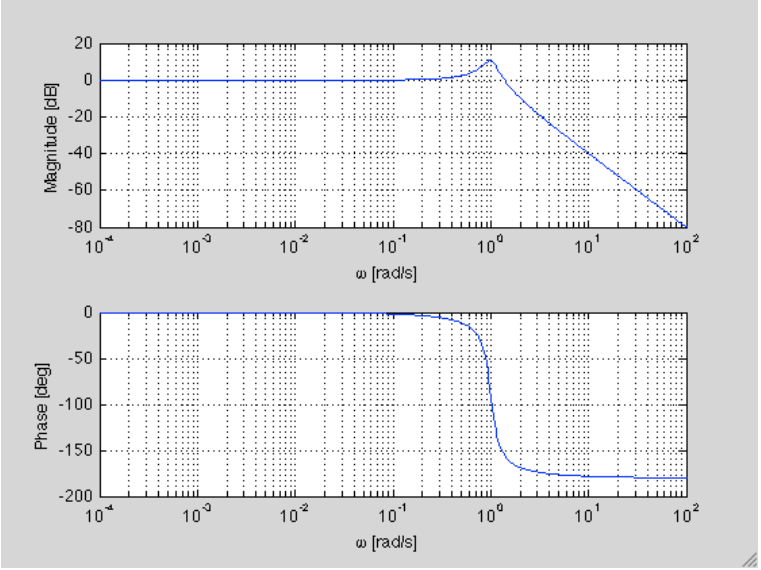
Diagrammi di Bode

Provare:

$$G(s) = \frac{1}{1 + 10s}$$



$$G(s) = \frac{1}{1 + 0.3s + s^2}$$

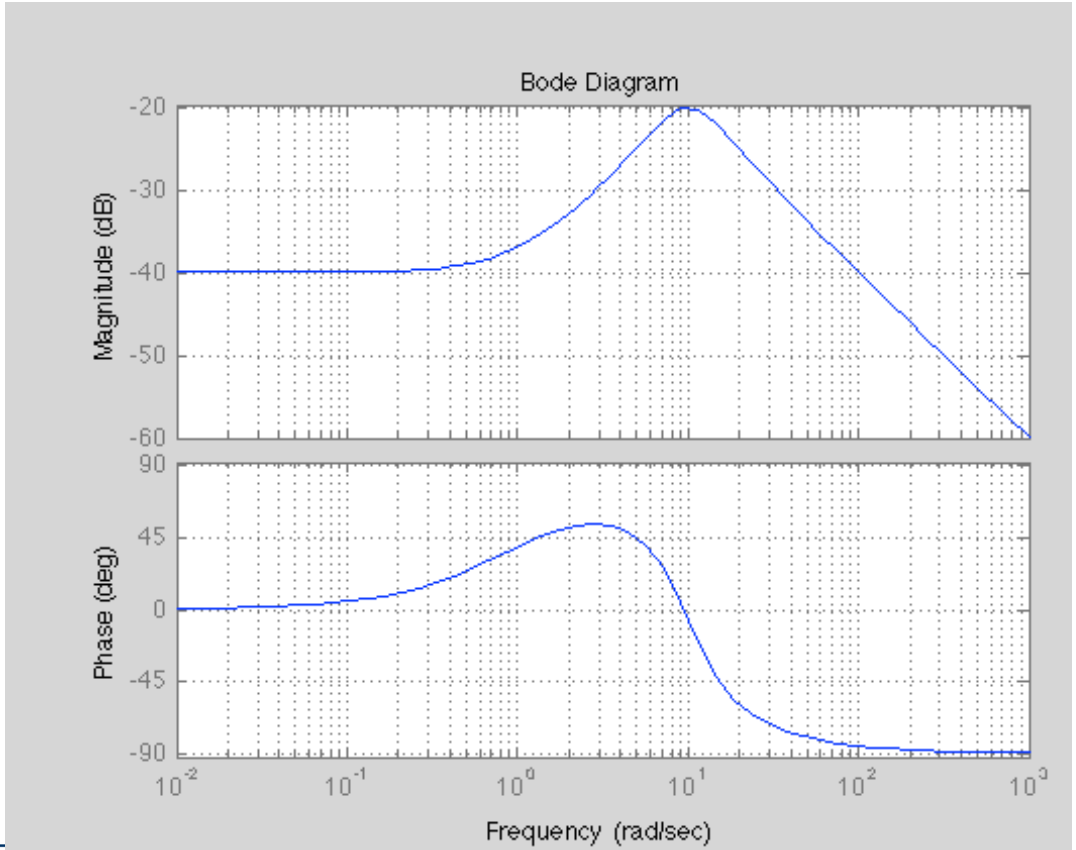


Comando bode

I diagrammi di Bode si possono visualizzare in Matlab col comando bode (prende come argomenti i sistemi definiti con i comandi visti nella sessione precedente: ss o tf).

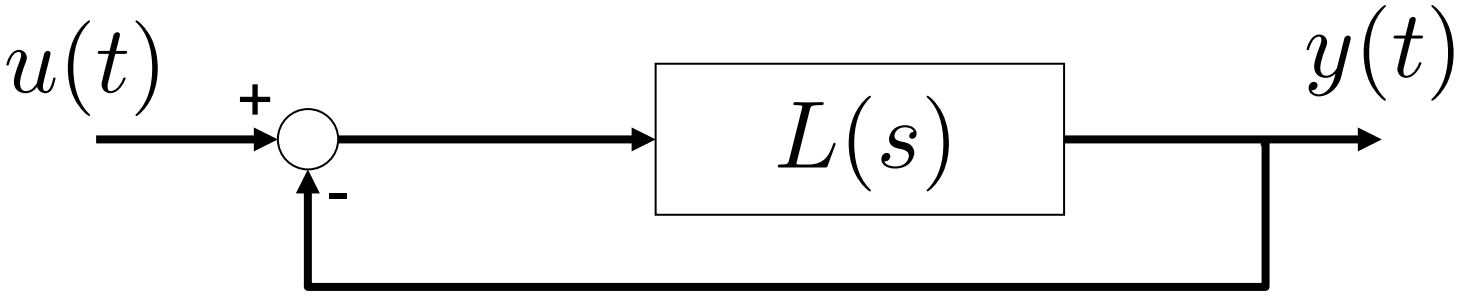
Esempio: per il sistema

```
>> sistema = tf([1 1],[1 10 100]);
>> bode(sistema);
```



Esercizio

Visualizziamo i diagrammi di Bode del sistema in figura:



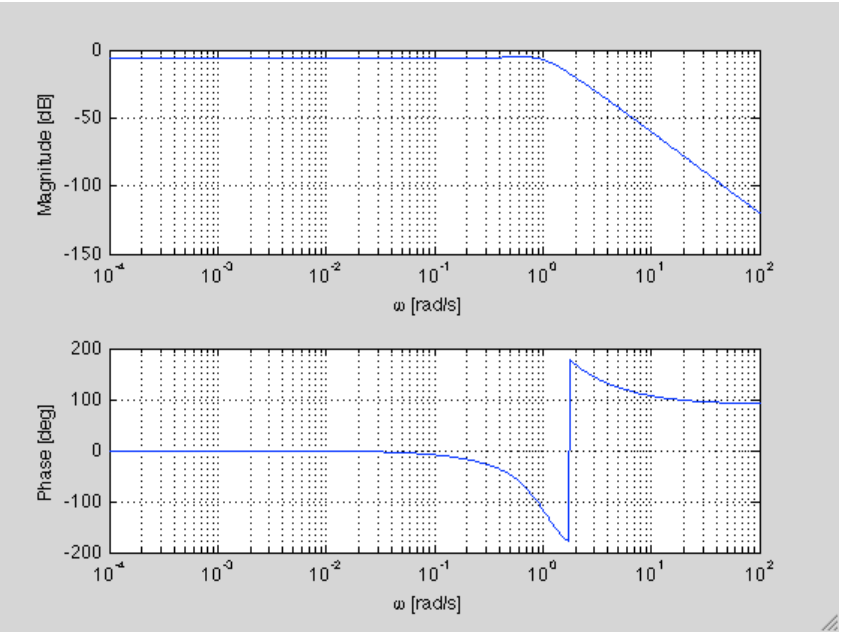
con $L(s) = \dfrac{1}{(1+s)^3}$.

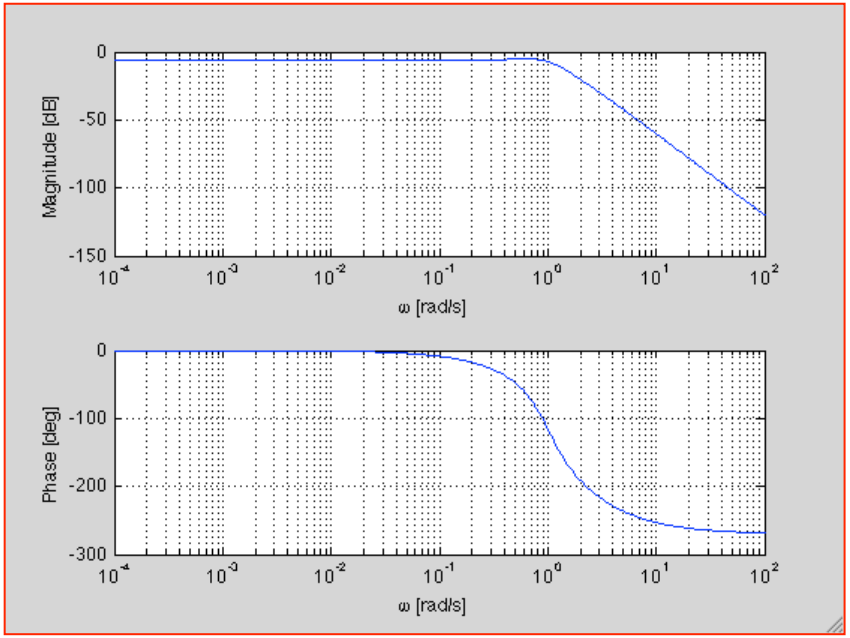
Step 1: calcolare (a mano) la funzione di trasferimento da u(t) a y(t) (***funzione di sensitività complementare***) e il suo guadagno.

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

Esercizio

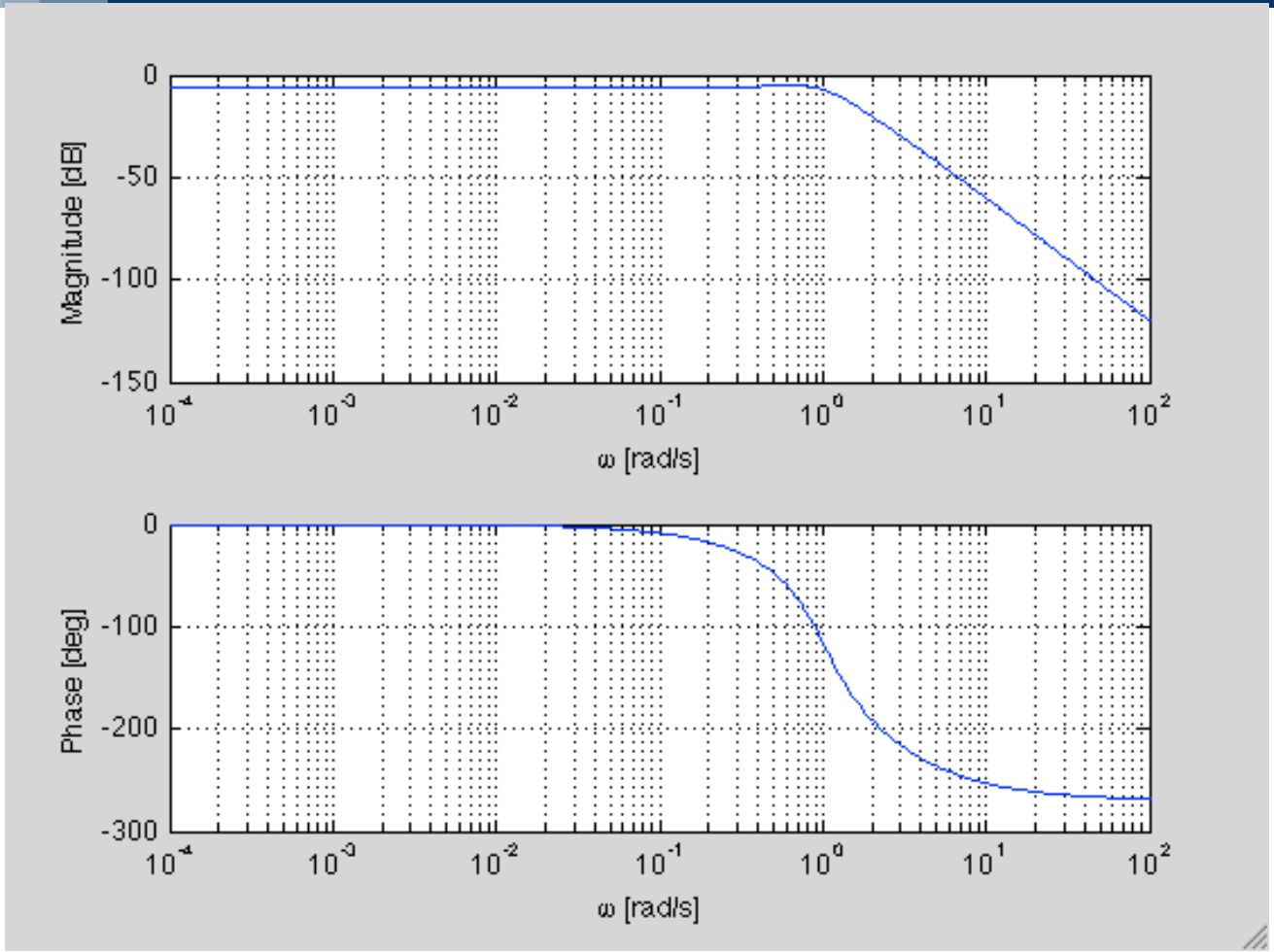
Step 2: visualizzare diagrammi di Bode della funzione di sensitività complementare. → Guadagno = ½.






```
>> subplot(2,1,2); semilogx(w,unwrap(faseR)*180/pi); grid on;
```

Esercizio

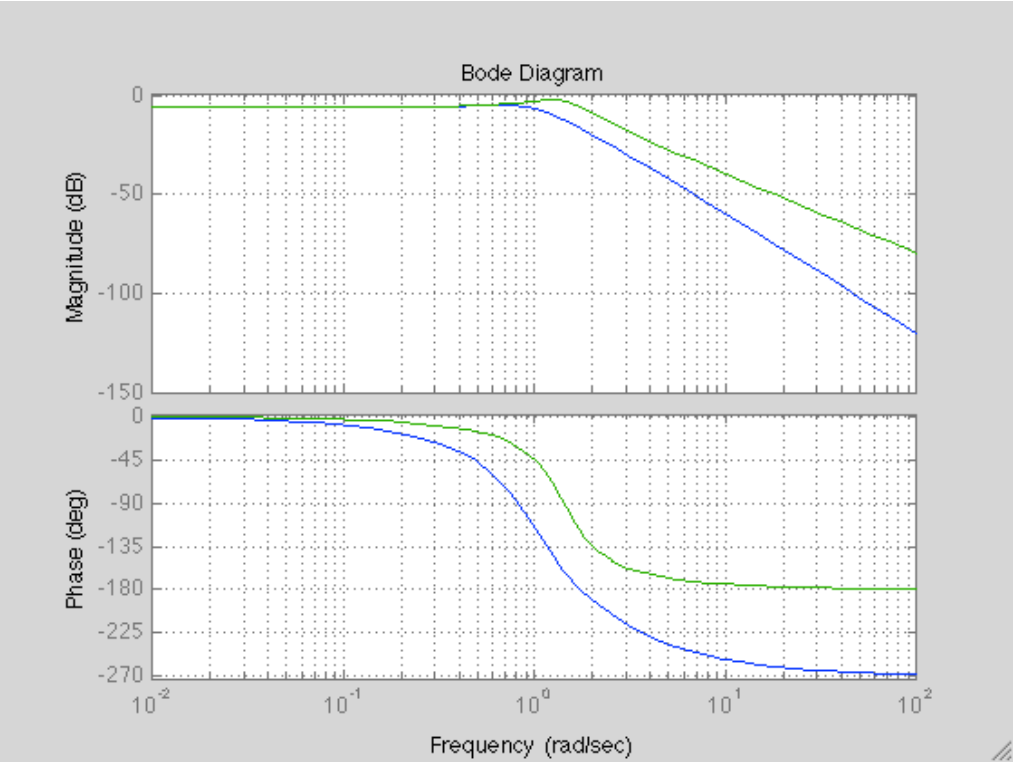


Piuttosto che una funzione del terzo ordine, possiamo descrivere (almeno a bassa frequenza) questo sistema con una funzione di trasferimento equivalente del secondo ordine?

Esercizio

Step 3: plottare sul grafico precedente la risposta in frequenza del sistema descritto dalla fdt:

$$G_{eq}(s) = \frac{1}{2 + s + s^2}$$



A bassa frequenza i due sistemi sono equivalenti!
Verifichiamo…

Teorema della risposta in frequenza

Se si applica a un sistema lineare asintoticamente stabile con funzione di trasferimento G(s) l'ingresso sinusoidale:

$$u(t) = U \sin(\omega_0 t)$$

l'uscita a transitorio esaurito assume l'andamento:

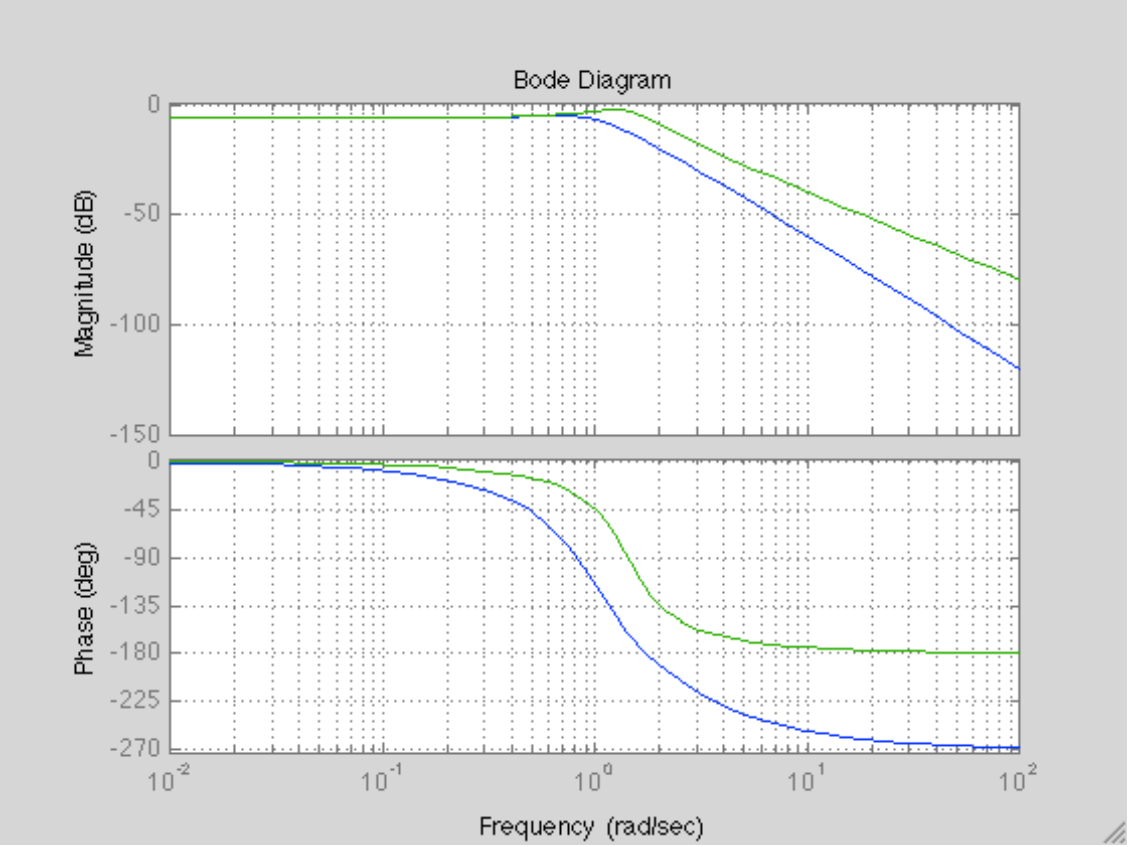
$$y(t) = |G(j\omega_0)|U \sin\left(\omega_0 t + \arg G(j\omega_0)\right)$$

Confrontiamo la risposta a un ingresso sinusoidale di ampiezza 1 e **pulsazione 0.01,0.1, 1** per i sistemi descritti da:

$$F(s) = \frac{1}{2 + 3s + 3s^2 + s^3} \quad G_{eq}(s) = \frac{1}{2 + s + s^2}$$

Teorema della risposta in frequenza

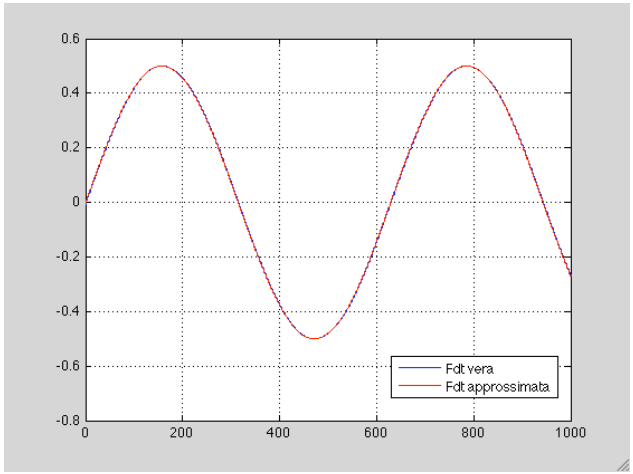
$$F(s) = \frac{1}{2 + 3s + 3s^2 + s^3} \quad G_{eq}(s) = \frac{1}{2 + s + s^2}$$



$$y(t) = |G(j\omega_0)|U \sin\left(\omega_0 t + \arg G(j\omega_0)\right)$$

Teorema della risposta in frequenza

Stesso guadagno a bassa frequenza, la fase cambia molto poco.

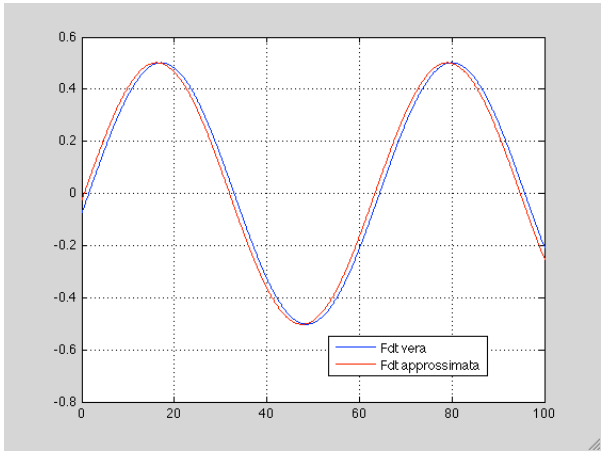



```
t = 0:0.1:1000;
num = 1;
dentrue = [1 3 3 2];
denapprox = [1 1 2];
j = sqrt(-1);
omega0 = 0.01;
u = sin(omega0*t);
R1 = polyval(num,j*omega0)/polyval(dentrue,j*omega0); % Risposta alla frequenza omega0
y1 = abs(R1)*1*sin(omega0*t+angle(R1));
R2 = polyval(num,j*omega0)/polyval(denapprox,j*omega0);
y2 = abs(R2)*1*sin(omega0*t+angle(R2));
figure
plot(t,y1,t,y2,'r')
grid on
legend('Fdt vera','Fdt approssimata');
```

$$y(t) = |G(j\omega_0)|U \sin\left(\omega_0 t + \arg G(j\omega_0)\right)$$

Teorema della risposta in frequenza

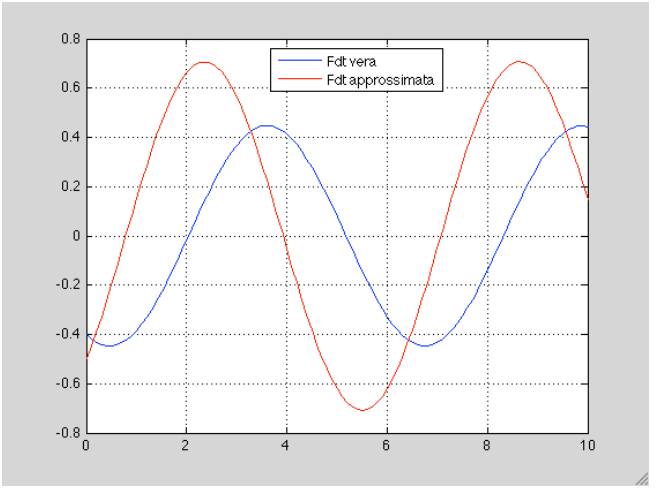
Stesso guadagno a bassa frequenza, la fase cambia un po' di più.




```
t = 0:0.1:100;
num = 1;
dentrue = [1 3 3 2];
denapprox = [1 1 2];
j = sqrt(-1);
omega0 = 0.1;
u = sin(omega0*t);
R1 = polyval(num,j*omega0)/polyval(dentrue,j*omega0);
y1 = abs(R1)*1*sin(omega0*t+angle(R1));
R2 = polyval(num,j*omega0)/polyval(denapprox,j*omega0);
y2 = abs(R2)*1*sin(omega0*t+angle(R2));
figure
plot(t,y1,t,y2,'r')
grid on
legend('Fdt vera','Fdt approssimata');
```

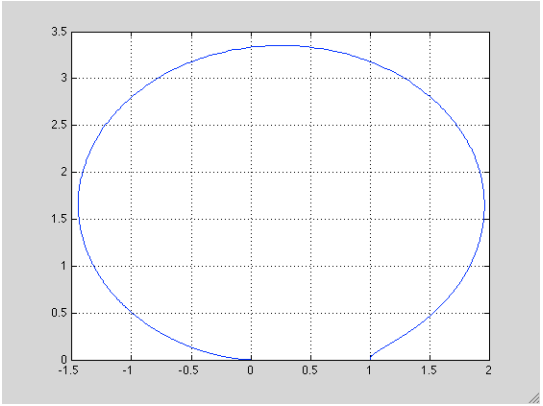
Teorema della risposta in frequenza

Aumentando la pulsazione della forzante l'approssimazione non è più valida! Sia il guadagno che la fase cambiano notevolmente!



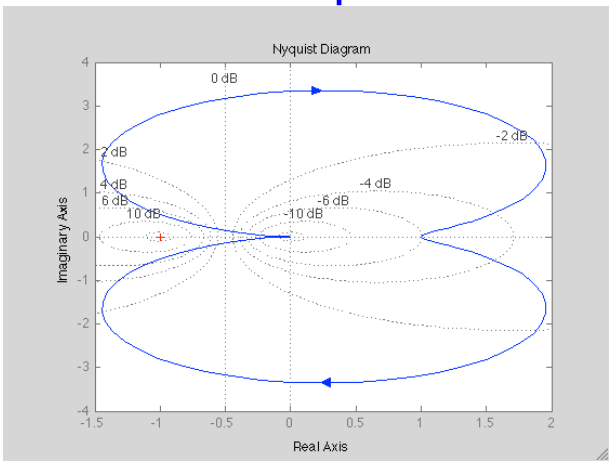

```
t = 0:0.1:10;
num = 1;
dentrue = [1 3 3 2];
denapprox = [1 1 2];
j = sqrt(-1);
omega0 = 1;
u = sin(omega0*t);
R1 = polyval(num,j*omega0)/polyval(dentrue,j*omega0);
y1 = abs(R1)*1*sin(omega0*t+angle(R1));
R2 = polyval(num,j*omega0)/polyval(denapprox,j*omega0);
y2 = abs(R2)*1*sin(omega0*t+angle(R2));
figure
plot(t,y1,t,y2,'r')
grid on
legend('Fdt vera','Fdt approssimata');
```

Diagramma polare



Il diagramma polare si disegna per punti come da definizione, o col comando nyquist.

```
num = 1; % numeratore
den = [1 0.3 1]; % denominatore
j = sqrt(-1); % unità immaginaria
w = logspace(-4, 4, 10000); % asse delle ascisse (pulsazioni)
R = polyval(num,j*w)./polyval(den,j*w); % calcolo effettivo della risposta in
    frequenza
modR = abs(R);
faseR = angle(R);
nyq = modR.*exp(-j*faseR);
plot(nyq),grid;
figure
```



nyquist(tf(num,den)) → Il comando nyquist ci dà il diagramma di Nyquist e la posizione del punto -1 in rosso.

Comandi utili

Riassumendo: abbiamo visto come si tracciano i diagrammi di Bode e di Nyquist.

I comandi Matlab sono:

>> bode

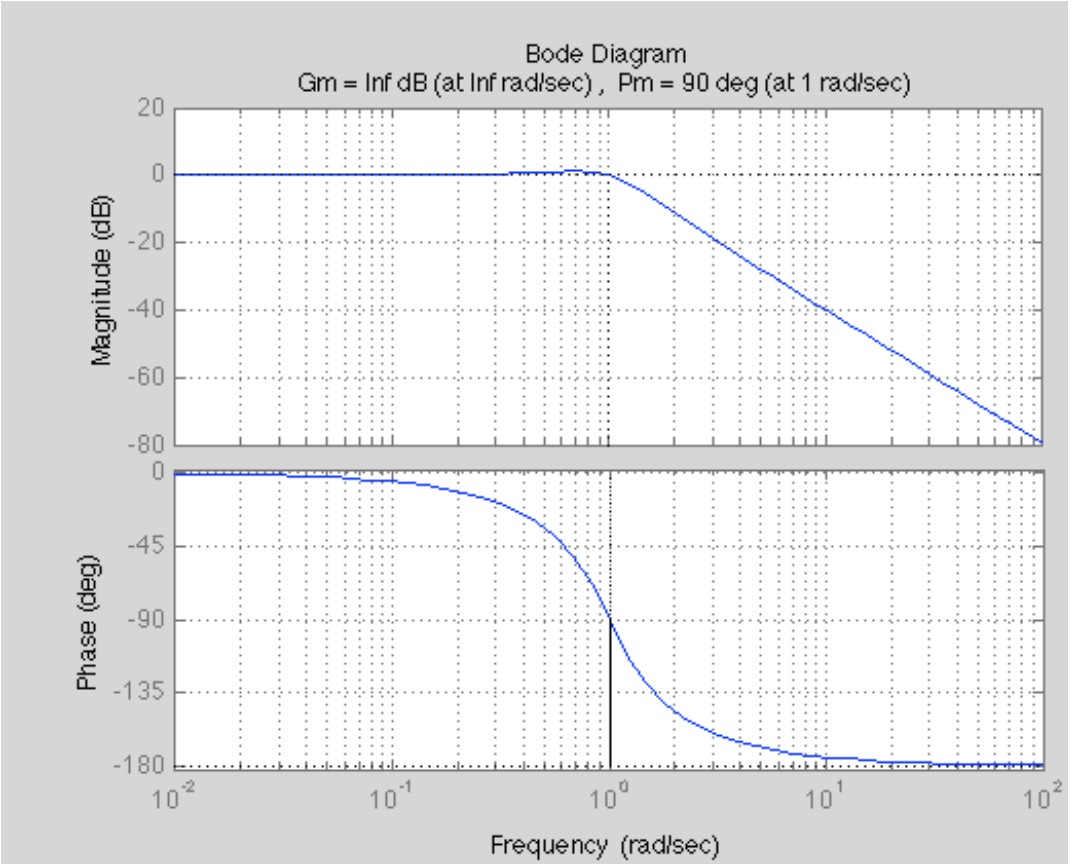
>> nyquist

>> margin

margin oltre ai diagrammi di Bode dà informazioni

quali il margine di fase

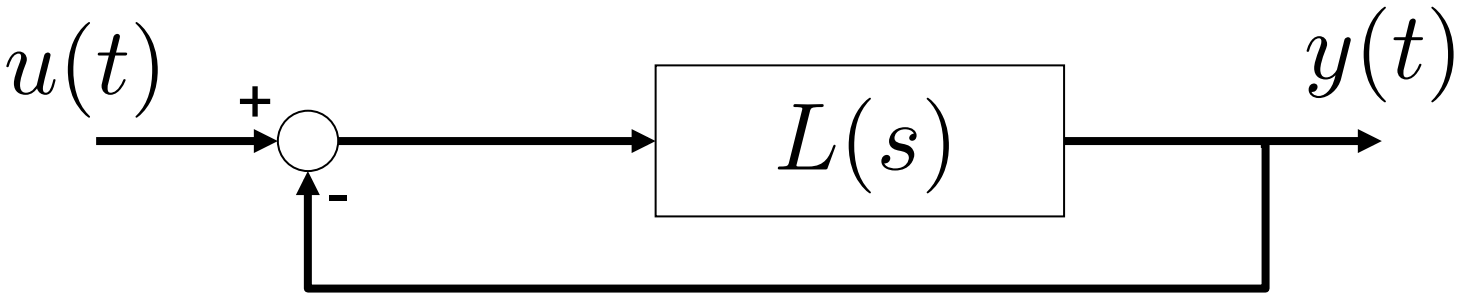
e il margine di guadagno.



→ provare margin(tf(1,[1 1 1]))

Domanda: perché c'è margine di guadagno infinito? Nyquist può aiutare a capire.

Effetti di un ritardo di tempo: stabilità al variare dei parametri



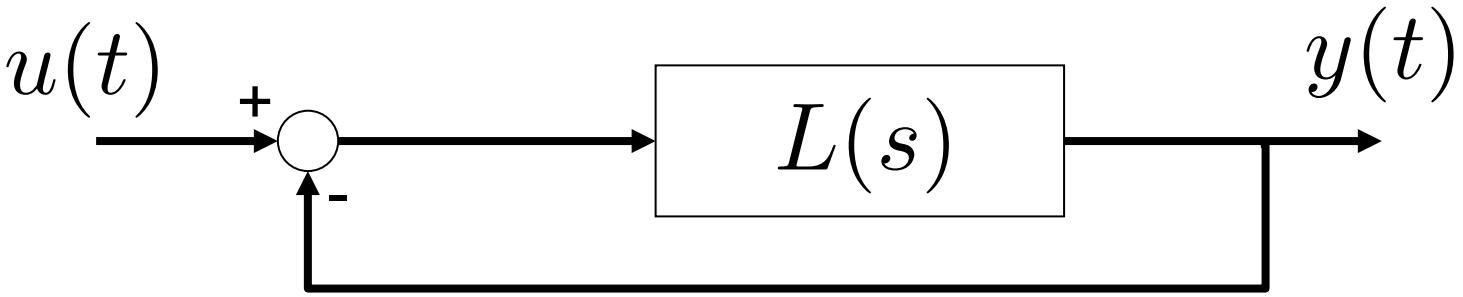
$$L(s) = \frac{\alpha}{(1+s)^3} e^{-\tau s}$$

Studiamo la stabilità del sistema col criterio di Nyquist al variare dei parametri $\alpha > 0$ e $\tau > 0$. Imponendo:

$$|L(jw)| = 1$$

$$\arg L(jw) = -\pi$$

Effetti di un ritardo di tempo: stabilità al variare dei parametri



$$L(s) = \frac{\alpha}{(1+s)^3} e^{-\tau s}$$

Si trovano le due equazioni (verificare per esercizio) seguenti, con $\tau^* > 0$ per $1 < \alpha < 8$.

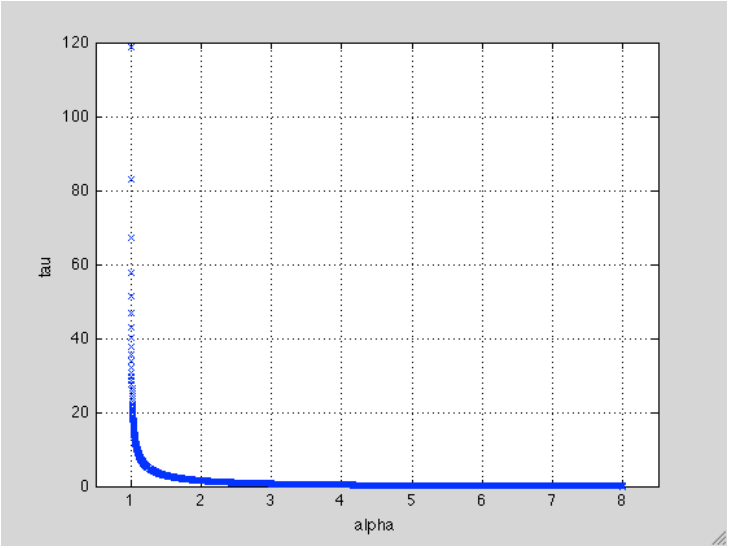
$$\omega^* = \sqrt{\alpha^{\frac{2}{3}} - 1} \qquad \tau^* = \frac{\pi - 3\arctan(\omega^*)}{\omega^*}$$

Effetti di un ritardo di tempo: stabilità al variare dei parametri

Disegniamo, per punti, il luogo geometrico descritto dalle equazioni seguenti. Al di sotto di questa curva, e per ritardi di tempo positivi, il sistema è asintoticamente stabile.

$$\omega^* = \sqrt{\alpha^{\frac{2}{3}} - 1} \qquad \tau^* = \frac{\pi - 3\arctan(\omega^*)}{\omega^*}$$

```
clear all, close all
alpha = 1:0.001:8;
omegastar = sqrt(alpha.^(2/3)-1);
taustar = (pi-3*atan(omegastar))./omegastar;
figure
plot(alpha,taustar,'x')
grid on
axis([0.5 8.5 0 120])
```



Effetti di un ritardo di tempo: verifica dei risultati con criterio di Nyquist

$$L(s) = \frac{\alpha}{(1+s)^3} e^{-\tau s}$$

Verifichiamo la stabilità osservando i diagrammi di Nyquist:

s = tf('s'); → Un modo alternativo per definire le funzioni di trasferimento

alpha = 8; → Provare diversi valori di alpha, tra 8 e 1.001

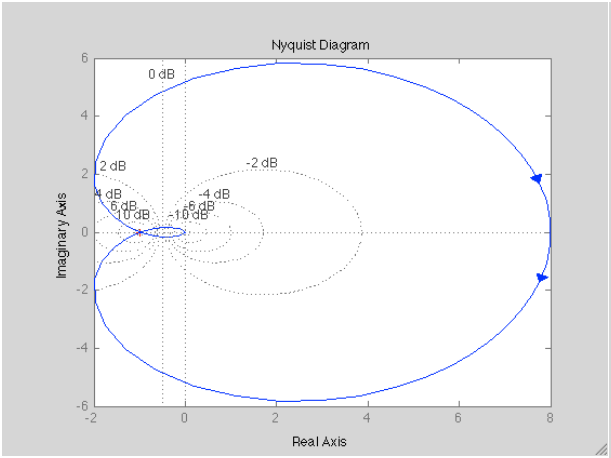
tau = (pi-3*atan(sqrt(alpha^(2/3)-1)))/(sqrt(alpha^(2/3)-1));

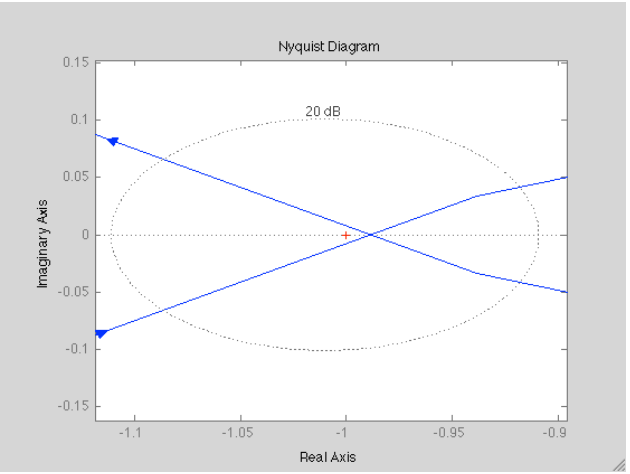
L = alpha/(1+s)^3;

G = L;

L.OutputDelay = tau;

nyquist(L)





Effetti di un ritardo di tempo: verifica dei risultati con criterio di Bode

$$L(s) = \frac{\alpha}{(1+s)^3} e^{-\tau s}$$

Verifichiamo la stabilità osservando i diagrammi di Nyquist:

s = tf('s'); → Un modo alternativo per definire le funzioni di trasferimento

alpha = 8; → Provare diversi valori di alpha, tra 8 e 1.001

tau = (pi-3*atan(sqrt(alpha^(2/3)-1)))/(sqrt(alpha^(2/3)-1));

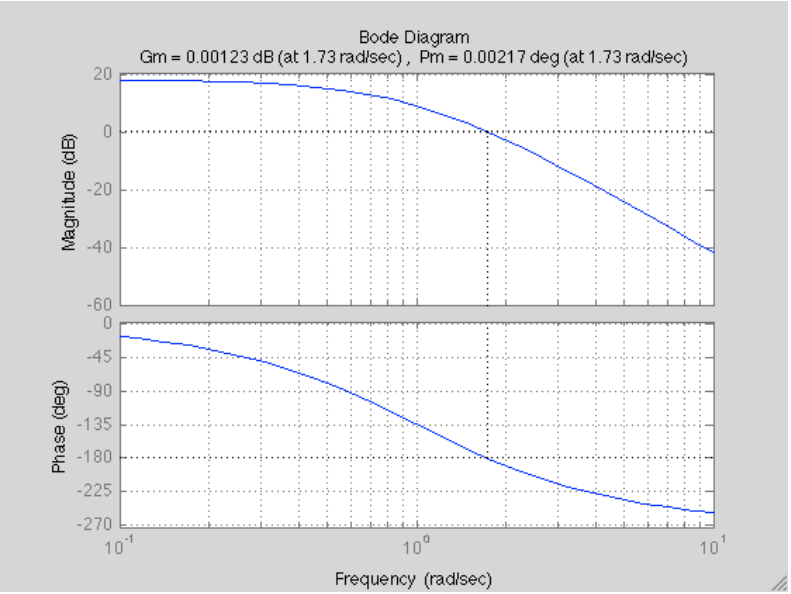
L = alpha/(1+s)^3;

G = L;

L.OutputDelay = tau;

margin(L)

NB: in alcuni casi margin dà margine di guadagno negativo dove il sistema dovrebbe essere stabile → help margin



Proprietà Bloccante degli Zeri

Il movimento dell'uscita in risposta a un ingresso esponenziale del tipo

$$u(t) = Ue^{\lambda t}$$

assume la forma (Teorema):

$$y(t) = G(\lambda)Ue^{\lambda t}$$

Ciò significa che se λ è uno zero di G(s), cioè $G(\lambda) = 0$, allora l'uscita forzata è nulla. Questo fatto va sotto il nome di ***proprietà bloccante degli zeri***, appunto perché gli zeri *bloccano* gli ingressi corrispondenti alle loro pulsazioni.

In generale se si vuole rendere insensibile una funzione di trasferimento a un ingresso particolare, si può pensare di aggiungere uno zero in corrispondenza della pulsazione di quell'ingresso (es: eliminare disturbi di rete a 50 Hz in un circuito di alimentazione elettrica).

Proprietà Bloccante degli Zeri

Fare qualche prova col codice seguente:

```
clear all, close all
t = 0:0.1:10;
omega0 = 1;
num = [1 (-omega0)]; ??? cosa cambia se si scrive num = [1 (-omega0+0.001)];
den = poly([-3 -5]);
j = sqrt(-1);
u = exp(omega0*t);
R = polyval(num,omega0)/polyval(den,omega0);
y = R.*exp(omega0*t);
plot(t,y)
grid on
```