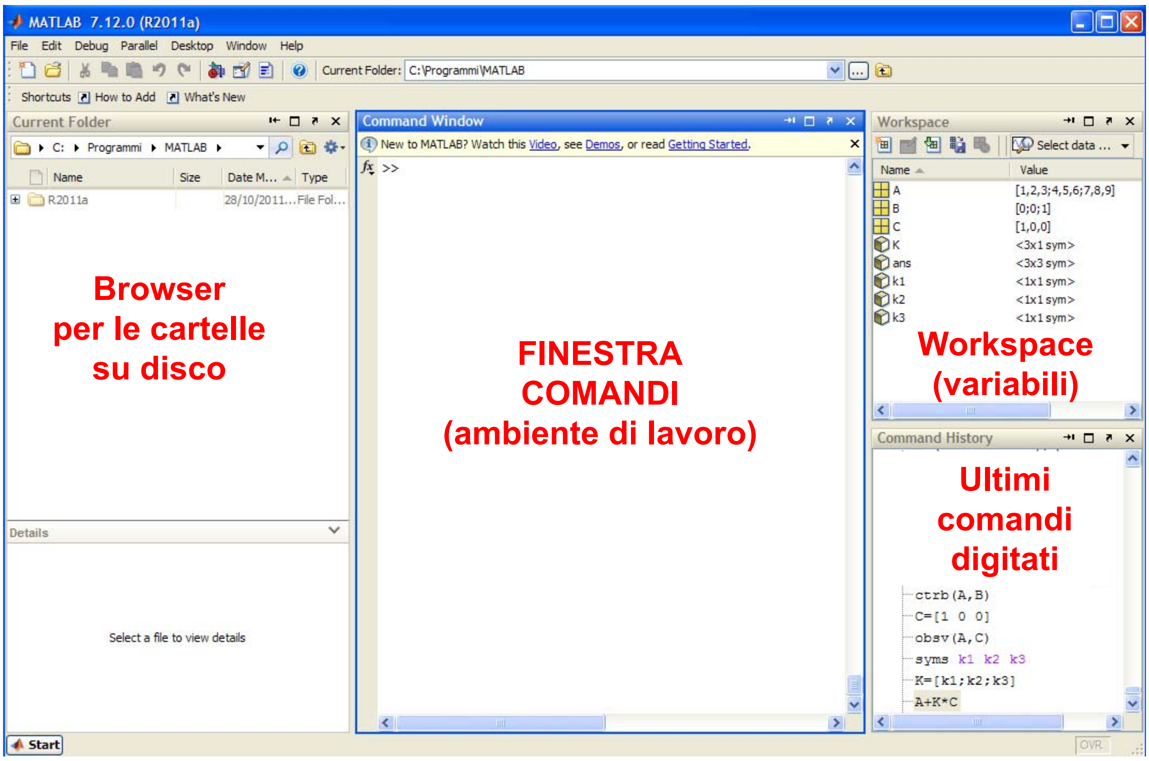
Fondamenti di Automatica

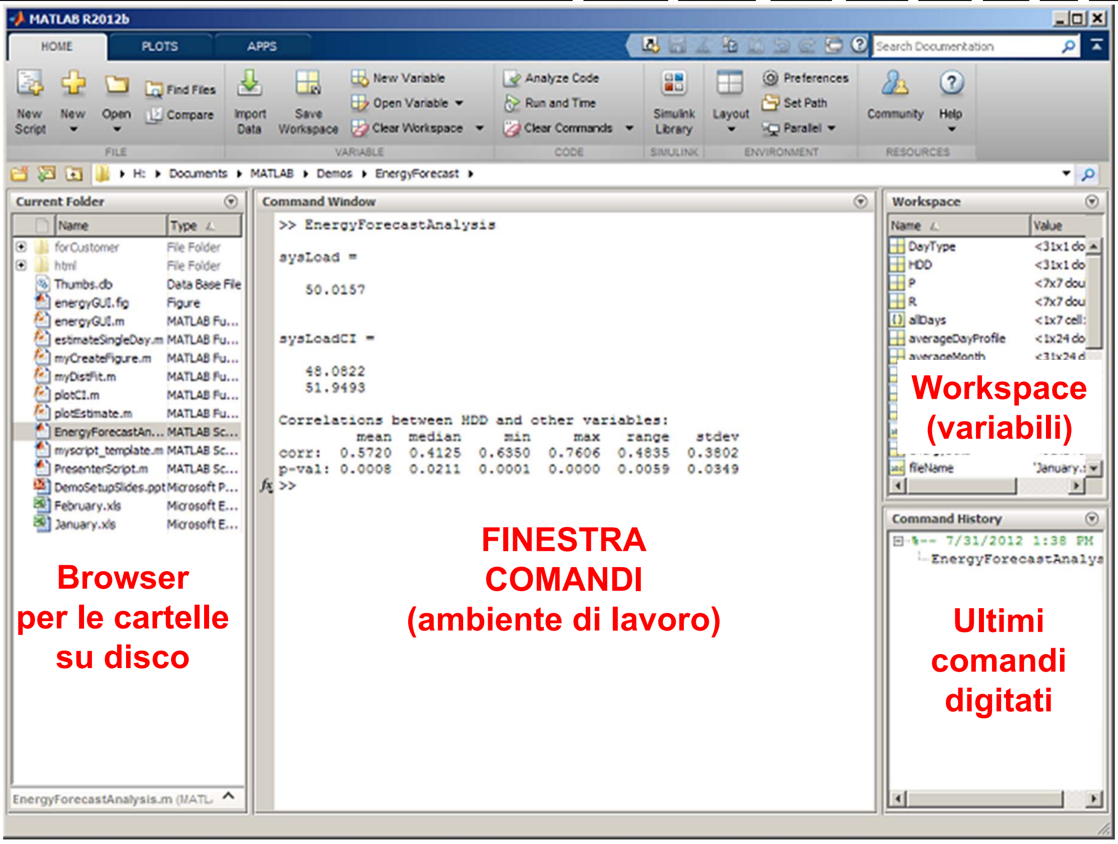
Cenni su Matlab
(e toolbox Control Systems + Symbolic)

Dott. Ing. Marcello Bonfè
Dipartimento di Ingegneria - Università di Ferrara
Tel. +39 0532 974839
E-mail: marcello.bonfe@unife.it

Matlab: interfaccia principale (fino a R2011a)



Matlab: interfaccia principale (da R2012b)



Matlab: definizione di variabili, vettori e matrici

Definire variabile scalare

`>> x = 3`

Definire vettore riga (1×3)

`>> x = [1 2 3]`

Idem, ma senza echo dell'output

`>> x = [1 2 3];`

Definire vettore colonna (3x1)

`>> x = [1; 2; 3]`

(oppure `>> x = [1 2 3]'`)

Definire matrice 3x4

`>> A = [1 2 3 4;5 6 7 8;9 10 11 12]`

Accedere / modificare elemento di riga 2 e colonna 1

`>> A(2,1) = 0`

Matlab: operazioni su matrici

- Le "solite" operazioni matematiche: `+,-,*,/,^`
- **Es.** >> `A^3` (potenza di matrice, solo se quadrata!)
- Precedute dal punto, sono eseguite elemento per elemento anziché in senso matriciale/vettoriale
- Operazioni specifiche per matrici / vettori:
  - Trasposta: `A'`
  - Determinante: `det(A)`
  - Inversa: `inv(A)`
  - Autovalori: `eig(A)`
  - Rango: `rank(A)`
  - Polinomio caratteristico: `poly(A)`
  - Esponenziale di matrice: `expm(A)`
  - Radici di un polinomio: `roots(x)` (x vettore dei coeff.)

Matlab: esponenziale di matrice (calcolo simbolico)

- In Matlab, è necessario definire la matrice A e il simbolo `t`, quest'ultima operazione possibile grazie al Symbolic Toolbox

```
>> A=[-4 0; 1 -4]
>> syms t
>> expm(A*t)
ans =
[ 1/exp(4*t),            0]
[ t/exp(4*t), 1/exp(4*t)]
```

NOTA: il risultato è simbolico, i termini esponenziali sono a denominatore, il che equivale ad esponente negativo

Matlab: esponenziale di matrice (calcolo simbolico)

- Nota la matrice esponenziale, è possibile calcolare il valore dello stato di un sistema dinamico noto lo stato iniziale e il tempo intercorso tra i due stati

```
>> x3=[1; 0]
>> x4=expm(A*(4-3))*x3
x4 =
    0.0183
    0.0183
```

NOTA: il risultato numerico equivale a e^{-4} (in Matlab `exp(-4)`) per entrambe le variabili di stato..

Matlab: test di controllabilità / osservabilità

- Grazie al Control Systems Toolbox, il test è eseguibile semplicemente lanciando i comandi:

```
>> P=ctrb(A,B)
```

per la matrice di raggiungibilità, poi ➔ `rank(P)` per il test di controllabilità

```
>> Qt=obsv(A,C)'
```

per la matrice di osservabilità, poi ➔ `rank(Qt)` per il test di osservabilità

Matlab: progetto analitico di controllo

Si consideri l'esempio da FdA-SolutionsGuide_2017.pdf, pagina 9:

$$A = \begin{bmatrix} -2 & 0 & 0 \\ 0 & 0 & -3 \\ 0 & 2 & -4 \end{bmatrix} \qquad B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

In Matlab:

```
>> A=[-2 0 0; 0 0 -3; 0 2 -4]
>> B=[1 0 0]'
```

E si supponga di voler progettare una retroazione stato ingresso ***u* = *H x***, al fine cioè di assegnare gli autovalori della matrice ***A+BH***

Matlab: progetto analitico di controllo

Il progetto richiede di definire la matrice ***H*** di dimensione 3x1: $\boldsymbol{H = [h_1\ h_2\ h_3]}$

In Matlab:

```
>> syms h1 h2 h3
>> H=[h1 h2 h3]
>> Acl=A+B*H
Acl =
[ h1 - 2, h2, h3]
[      0,  0, -3]
[      0,  2, -4]
```

Matlab: progetto analitico di controllo

Per capire anzitutto quanti autovalori si possono assegnare e in relazione a quali coefficienti di ***H***, si deve determinare il polinomio caratteristico della matrice del sistema ad anello chiuso.

In Matlab:

```
>> syms lambda
>> polycar=det(lambda*eye(3)-Acl)
polycar =
14*lambda - 6*h1 - 4*h1*lambda - h1*lambda^2
+ 6*lambda^2 + lambda^3 + 12
```


Matlab: progetto analitico di controllo

ATTENZIONE: il polinomio caratteristico ottenuto a questo punto potrebbe NON avere tutti gli autovalori dipendenti dai coefficienti di ***H***. In effetti, il numero di autovalori assegnabili è pari al rango della matrice di raggiungibilità (di osservabilità nel caso di progetto di osservatori).

TUTTAVIA, non è strettamente necessario avere già svolto l'analisi di raggiungibilità (o osservabilità) per eseguire il progetto sul sottoinsieme opportuno degli autovalori assegnabili

Matlab: progetto analitico di controllo

Ci sono due modi per evidenziare questo aspetto.

Primo modo:

```
>> eig(Acl)
ans =
 - 2 - 2^(1/2)*i
 - 2 + 2^(1/2)*i
          h1 - 2
```

← autovalore assegnabile

Nell'esempio specifico, l'unico autovalore assegnabile è quello in **h1 - 2**

Matlab: progetto analitico di controllo

Secondo modo:

```
>> factor(polycar)
ans =
-(lambda^2 + 4*lambda + 6)*(h1 - lambda - 2)
```

NOTA: questo secondo modo è quello più utile per completare il progetto una volta fissati i valori desiderati degli autovalori, soprattutto nel caso in cui gli autovalori assegnabili siano più di uno!!

Matlab: progetto analitico di controllo

INFATTI: si consideri ora la stessa matrice ***A*** precedente ma con:

```
>> B=[1 0 0]'
>> Acl=A+B*H
>> polycar=det(lambda*eye(3)-Acl)
>> factor(polycar)
ans =
(lambda + 2)*(4*lambda - 2*h3 - 4*h2 -
h2*lambda + lambda^2 + 6)
```


Matlab: progetto analitico di controllo

A questo punto risultano due autovalori assegnabili, poiché il polinomio nel quale compaiono i coefficienti di ***H*** è di ordine 2. Per determinare tali coefficienti, occorre fissare i valori desiderati per gli autovalori e calcolare il corrispondente polinomio caratteristico desiderato. Ad esempio con autovalori in -3 e -4:

```
>> polydes=(lambda+3)*(lambda+4)
>> expand(polydes)
ans =
lambda^2 + 7*lambda + 12
```

Matlab: progetto analitico di controllo

Per concludere, occorre uguagliare i coefficienti di pari grado tra **`polydes`** e il termine raccolto in precedenza da **`polycar`** con l'operazione **`factor`**

L'operazione va fatta copiando manualmente tali coefficienti e inserendoli in un sistema di N equazioni:

```
>>[h2 h3]=solve('4-h2=7',
                '6-2*h3-4*h2=12')

h2 = -3

h3 = 3
```


Matlab: progetto analitico di controllo

CONTROPROVA:

```
>>H=[0 -3 3]
```
← **h1 è arbitrario.. 0 vale come ogni altro numero**

```
>>Acl=A+B*H

>>eig(Acl)

ans =
```

`-2` ← **NON modificato dal progetto!**

`-4` ← **assegnato da H**

`-3` ← **assegnato da H**

Matlab: progetto analitico di controllo

NOTA: nel caso in cui l'operazione non riveli nessuna possibile fattorizzazione in termini più semplici, il polinomio caratteristico desiderato deve essere di pari grado rispetto a quello del polinomio caratteristico del sistema chiuso in retroazione.

ESEMPIO: diversa matrice ***A***:

```
>> A=[-2 0 0; 0 0 -3; 0 2 -4]
>> B=[1 0 0]'
```


Il progetto diventa:

```
>> syms h1 h2 h3
>> H=[h1 h2 h3]
>> Acl=A+B*H
Acl =
[ h1 - 2, h2, h3]
[      1,  0, -3]
[      0,  2, -4]
```

Matlab: progetto analitico di controllo

```
>> polycar=det(lambda*eye(3)-Acl)
polycar =
14*lambda - 4*h2 - 2*h3 - 6*h1 -
4*h1*lambda - h2*lambda - h1*lambda^2
+ 6*lambda^2 + lambda^3 + 12

>> factor(polycar)
ans =
14*lambda - 4*h2 - 2*h3 - 6*h1 -
4*h1*lambda - h2*lambda - h1*lambda^2
+ 6*lambda^2 + lambda^3 + 12
```


Matlab: progetto analitico di controllo

Supponendo di voler ottenere autovalori in -3, -4 e -5:

```
>>polydes=(lambda+3)*(lambda+4)*(lambda+5)
>> expand(polydes)
ans =
lambda^3 + 12*lambda^2 + 47*lambda + 60
>> [h1 h2 h3]=solve('6-h1=12',
    '14-4*h1-h2=47','12-6*h1-4*h2-2*h3=60')
h1 = -6
h2 = -9
h3 = 12
```

Matlab: progetto analitico di controllo

CONTROPROVA:

```
>>H=[-6 -9 12]
>> Acl=A+B*H
>> eig(Acl)
ans =
   -5.0000  ← tutti assegnati da H
   -4.0000  ← tutti assegnati da H
   -3.0000  ← tutti assegnati da H
```


Matlab: trasformate e antitrasformate di Laplace

- Il Symbolic Toolbox contiene le funzioni per il calcolo simbolico (appunto) delle trasformate di Fourier, Laplace, Z e relative inverse
- Ad esempio, si consideri la funzione di trasferimento:

$$G(s) = \frac{5s+12}{(s^2+5s+6)}$$

e si supponga di dover trovare la funzione del tempo per la sua risposta al gradino, cioè l'antitrasformata di:

$$Y(s) = G(s)U(s) = G(s)\frac{1}{s}$$

- La soluzione manuale richiede l'applicazione del metodo di scomposizione in fratti semplici (v. slide 4-12 in FdA-2.2-RispostaSistemiElementari_2017.pdf)

Matlab: trasformate e antitrasformate di Laplace

- In Matlab:

```
>> syms s
>> G=(5*s+12)/(s^2 + 5*s + 6)
G =
(5*s + 12)/(s^2 + 5*s + 6)
>> U = 1/s
>> Y = G*U
>> y = ilaplace(Y)
y =
2 - 1/exp(3*t) - 1/exp(2*t)
```

NOTA: i poli della funzione di trasferimento sono -2 e -3

Matlab: trasformate e antitrasformate di Laplace

- Ovviamente, il Symbolic Toolbox potrebbe anche essere usato per calcolare esplicitamente il passaggio da una rappresentazione nello spazio degli stati (matrici A,B,C,D) alla corrispondente funzione (o matrice) di trasferimento
- Tuttavia, essendo tale operazione tipicamente necessaria nel progetto di sistemi di controllo, il Control Systems Toolbox contiene funzioni specifiche, basate su strutture dati ancora più specifiche

Matlab: trasformate e antitrasformate di Laplace

- Si considerino le matrici:

$$A = \begin{bmatrix} -3 & 0 \\ 1 & -6 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 1 \end{bmatrix}$$

- In Matlab:

```
>> A=[-3 0; 1 -6]
>> B=[1; 1]
>> C=[1 1]
```


Matlab: trasformate e antitrasformate di Laplace

- Soluzione 1 (Symbolic Toolbox):

```
>> syms s
>> sA=inv(s*eye(2) - A)
```

← **eye(2)= identità 2x2..**

```
>> G=C*sA*B
G =
1/(s + 3) + 1/(s + 6) + 1/((s + 3)*(s + 6))
>> G=collect(G)
G =
(2*s + 10)/(s^2 + 9*s + 18)
```

Matlab: trasformate e antitrasformate di Laplace

- Soluzione 2 (Control Systems Toolbox):

```
>> sys=ss(A,B,C,0)
```
← D=0, necessario quarto parametro..

```
>> G=tf(sys)
Transfer function:
   2 s + 10
--------------
s^2 + 9 s + 18
```

Oppure anche, calcolando i coefficienti di numeratore e denominatore della FdT:

```
>> [N,D]=ss2tf(A,B,C,0)
N = 0      2     10
D = 1      9     18
>> G=tf(N,D)
```


Matlab: diagrammi di Bode e luogo delle radici

- Oltre al passaggio alla funzione **tf(num,den)** dei due vettori contenenti i coefficienti della FdT, esiste un'alternativa comoda per definire la FdT con la struttura del Control Systems Toolbox:

```
>> s=tf('s')
```
← "definisce" la variabile di Laplace

```
>> G=10*(1+s)^2/s/(1+s/0.1)/(1+s/100)
```

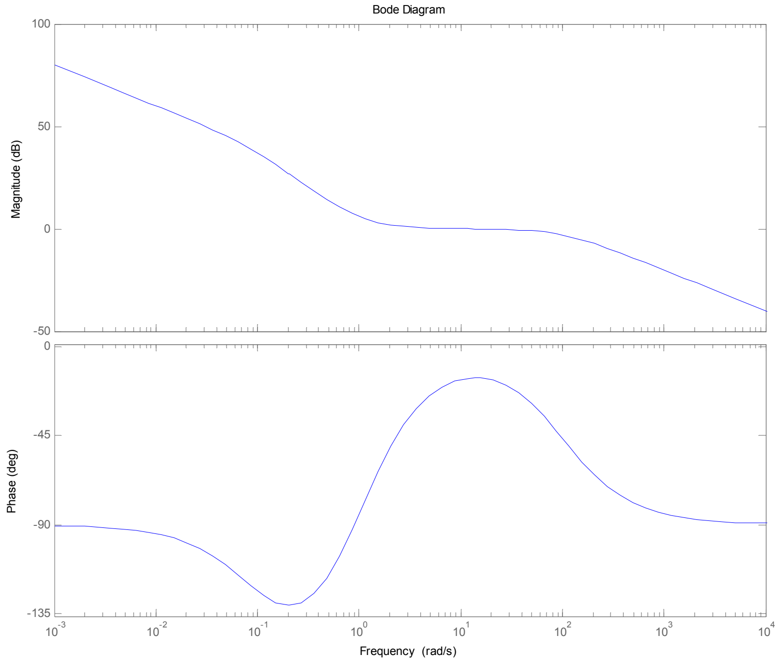
NOTA1: l'esempio è tratto da FdA-SolutionsGuide_2017.pdf, pagina 24-25

NOTA2: **in questo caso s NON è una variabile simbolica**, ma una vera e propria FdT rappresentata con la struttura dati corrispondente del Control Systems Toolbox..

Matlab: diagrammi di Bode e luogo delle radici

- Una volta definita la FdT, è immediato visualizzare il corrispondente diagramma di Bode (ESATTO!):

```
>> bode(G)
```

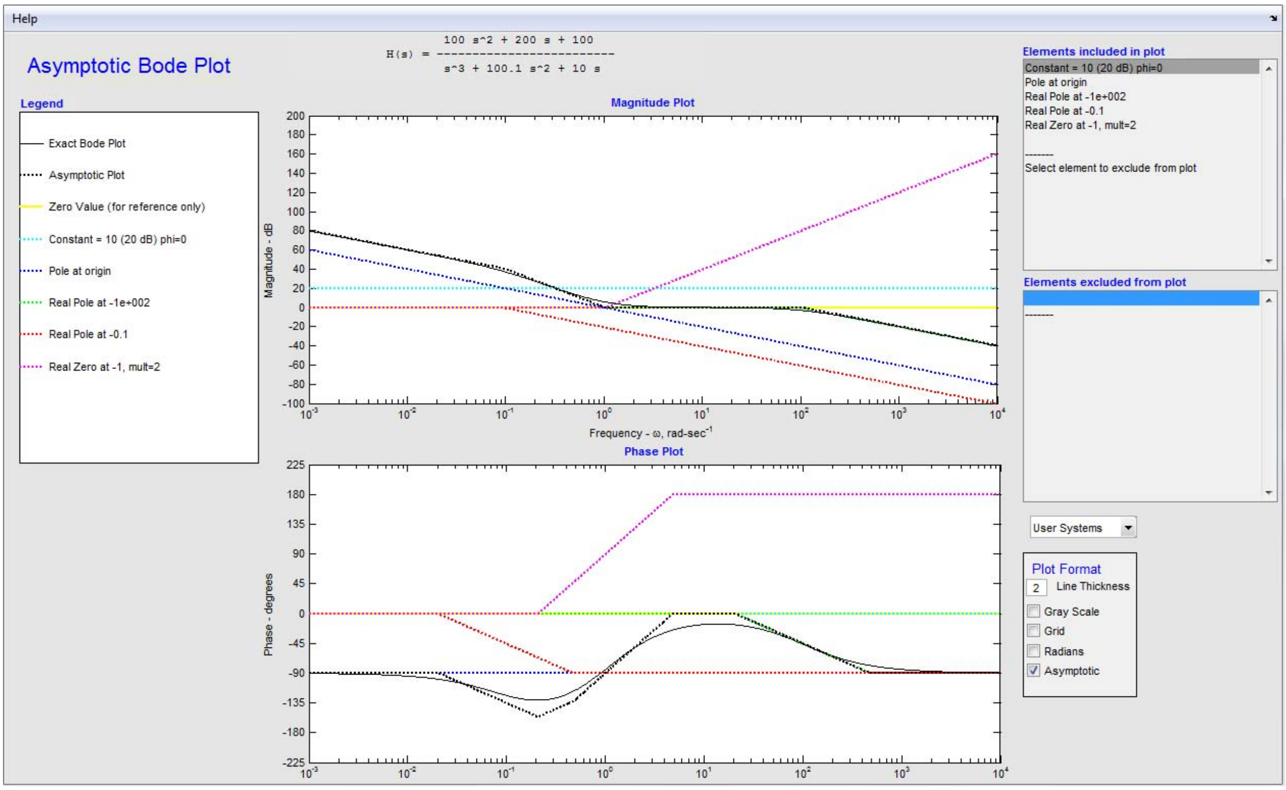


Matlab: diagrammi di Bode e luogo delle radici

- Nel Control Systems Toolbox di Matlab non esistono funzioni per visualizzare i diagrammi di Bode asintotici (i.e. approssimati) introdotti comunemente nei corsi di base di Automatica
- Peraltro, il metodo di tracciamento dei diagrammi di Bode con approssimazioni basate su linee spezzate è appunto pensato per un rapido svolgimento manuale, qualitativo
- Tuttavia, molti siti didattici propongono funzioni Matlab per confrontare il diagramma esatto e quello approssimato, tra le quali la più interessante è stata reperita da: **http://lpsa.swarthmore.edu/Bode/BodePlotGui.html**

  e **leggermente adattata per questa presentazione**
- `>> BodePlotGui(G)`

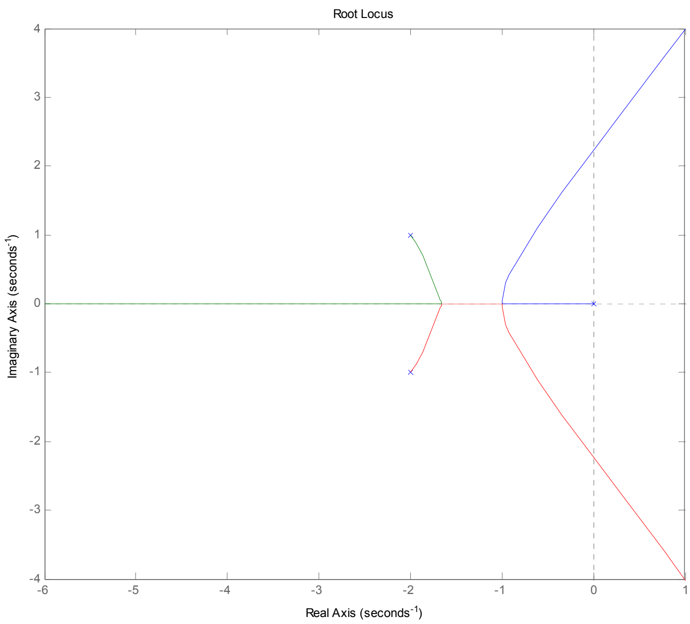
Matlab: diagrammi di Bode e luogo delle radici



Matlab: diagrammi di Bode e luogo delle radici

- Analogamente, è possibile visualizzare immediatamente il luogo delle radici

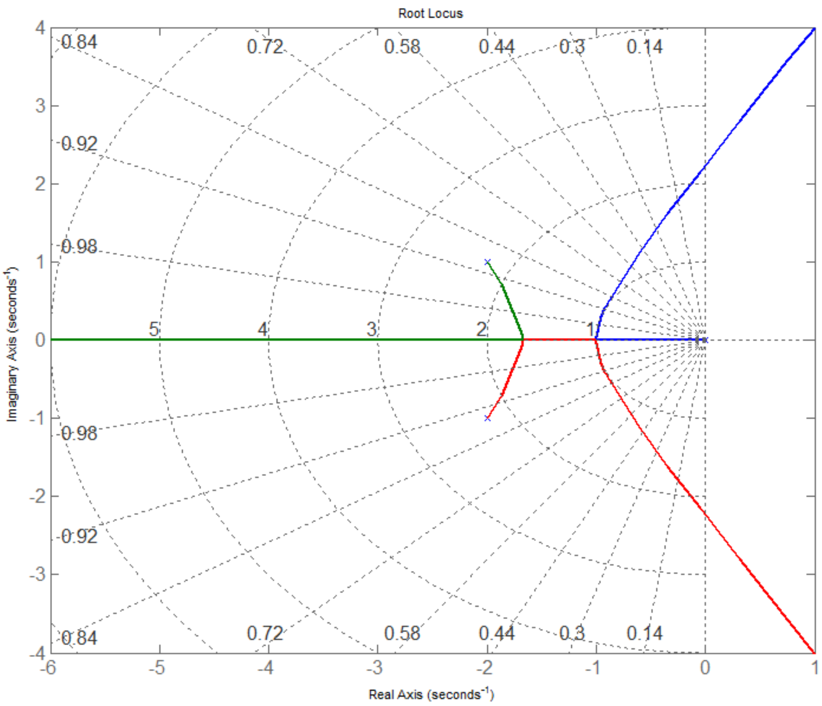
```
>> G=1/s/(s^2+4*s+5)
Transfer function:
        1
-----------------
s^3 + 4 s^2 + 5 s
>> rlocus(G)
```



Matlab: diagrammi di Bode e luogo delle radici

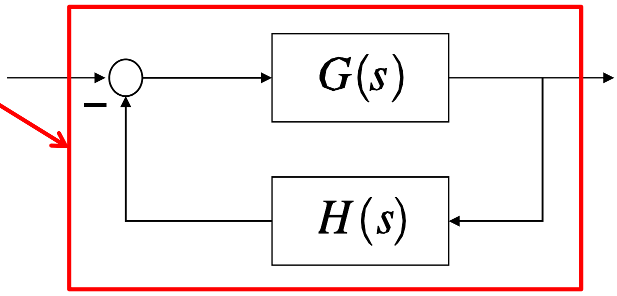
- **NOTA**: abilitando la *grid* (tramite menu via tasto destro del mouse) vengono visualizzati:
  - le circonferenze sulle quali giacciono i poli complessi coniugati con la stessa pulsazione naturale ω_n
  - i raggi sui quali giacciono poli complessi coniugati con lo stesso coefficiente di smorzamento δ
  - ovviamente, solo per il semipiano sinistro, corrispondente alla regione in cui devono giacere i poli per funzioni di trasferimento asintoticamente stabili..

Matlab: diagrammi di Bode e luogo delle radici



Matlab: altre funzioni del Control Systems Toolbox

```
>> Gcl = feedback(G,H)
```



`>> Gp = parallel(G1,G2)` ← parallelo di FdT

`>> Gs = series(G1,G2)` ← serie di FdT

`>> step(G)` ← grafica la risposta al gradino

`>> impulse(G)` ← grafica la risp. impulsiva

Matlab: criterio di Routh, errori a regime, ecc..

PURTROPPO, non è possibile mescolare elementi del Symbolic Toolbox con quelli del Control Systems Toolbox (es. FdT con coefficienti simbolici..)

PERTANTO, esercizi come quelli proposti all'esame su:

- riduzione di diagrammi a blocchi
- determinazione di intervalli di stabilità per sistemi in retroazione (criterio di Routh)
- calcolo di coefficienti t.c. si abbia
  - un certo errore a regime
  - oppure, una certa pulsazione naturale ω_n, coefficiente di smorzamento δ, tempo di assestamento T_a, ecc.

risultano in genere più articolati (o impossibili) da risolvere con l'ausilio di Matlab, piuttosto che manualmente, pertanto non verranno trattati in questa introduzione..

CENNI SU MATLAB

FINE