Appendix A
Matlab Control Toolbox Functions

Purpose Bode frequency response of LTI systems

Syntax

```
bode(sys)
bode(sys,w)

bode(sys1,sys2,...,sysN)
bode(sys1,sys2,...,sysN,w)
bode(sys1,'PlotStyle1',...,sysN,'PlotStyleN')

[mag,phase,w] = bode(sys)
```

Description

bode computes the magnitude and phase of the frequency response of LTI systems. When invoked without left-hand arguments, bode produces a Bode plot on the screen. Bode plots are used to analyze system properties such as the gain margin, phase margin, DC gain, bandwidth, disturbance rejection, and stability.

bode(sys) plots the Bode response of an arbitrary LTI model sys. This model can be continuous or discrete, and SISO or MIMO. In the MIMO case, bode produces an array of Bode plots, each plot showing the Bode response of one particular I/O channel. The frequency range is determined automatically based on the system poles and zeros.

bode(sys,w) explicitly specifies the frequency range or frequency points to be used for the plot. To focus on a particular frequency interval [wmin,wmax], set w = {wmin,wmax}. To use particular frequency points, set w to the vector of desired frequencies. Use logspace to generate logarithmically spaced frequency vectors. All frequencies should be specified in radians/sec.

bode(sys1,sys2,...,sysN) or bode(sys1,sys2,...,sysN,w) plots the Bode responses of several LTI models on a single figure. All systems must have the same number of inputs and outputs, but may otherwise be a mix of continuous and discrete systems. This syntax is useful to compare the Bode responses of multiple systems.

bode(sys1,'PlotStyle1',...,sysN,'PlotStyleN') further specifies which color, linestyle, and/or marker should be used to plot each system. For example,

```
bode(sys1,'r--',sys2,'gx')
```

uses red dashed lines for the first system sys1 and green 'x' markers for the second system sys2.

When invoked with left-hand arguments,

```
[mag,phase,w] = bode(sys)
[mag,phase] = bode(sys,w)
```

return the magnitude and phase (in degrees) of the frequency response at the frequencies w (in rad/sec.). The outputs mag and phase are 3-D arrays with the frequency as the last dimension (see "Arguments" below for details). You can convert the magnitude to decibels by

```
magdb = 20*log10(mag).
```

Arguments

The output arguments mag and phase are 3-D arrays with dimensions

$$(\text{number of outputs}) \times (\text{number of inputs}) \times (\text{length of w})$$

For SISO systems, mag(1,1,k) and phase(1,1,k) give the magnitude and phase of the response at the frequency ω_k = w(k) :

$$\text{mag(1,1,k)} = |h(j\omega_k)|$$
$$\text{phase(1,1,k)} = \angle h(j\omega_k)$$

MIMO systems are treated as arrays of SISO systems and the magnitudes and phases are computed for each SISO entry h_{ij} independently (h_{ij} is the transfer function from input j to output i). The values mag(i,j,k) and phase(i,j,k) then characterize the response of h_{ij} at the frequency w(k):

$$\text{mag(i,j,k)} = |h_{ij}(j\omega_k)|$$
$$\text{phase(i,j,k)} = \angle h_{ij}(j\omega_k)$$

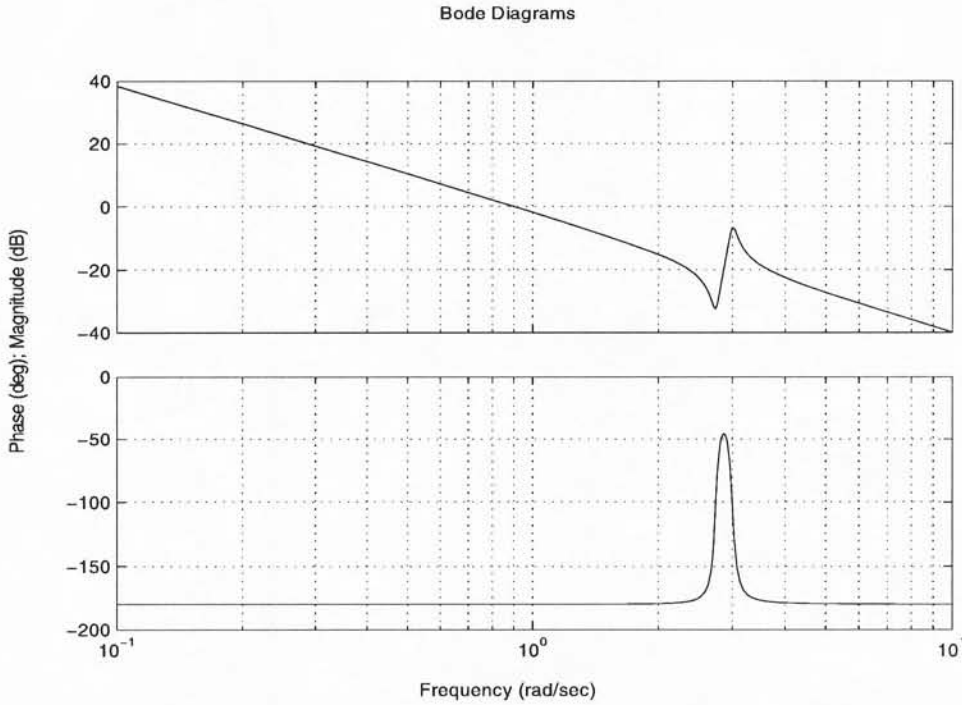
Example

You can plot the Bode response of the continuous SISO system

$$H(s) = \frac{s^2 + 0.1s + 7.5}{s^4 + 0.12s^3 + 9s^2}$$

by

```
» g = tf([1 0.1 7.5],[1 0.12 9 0 0]);
» bode(g)
```

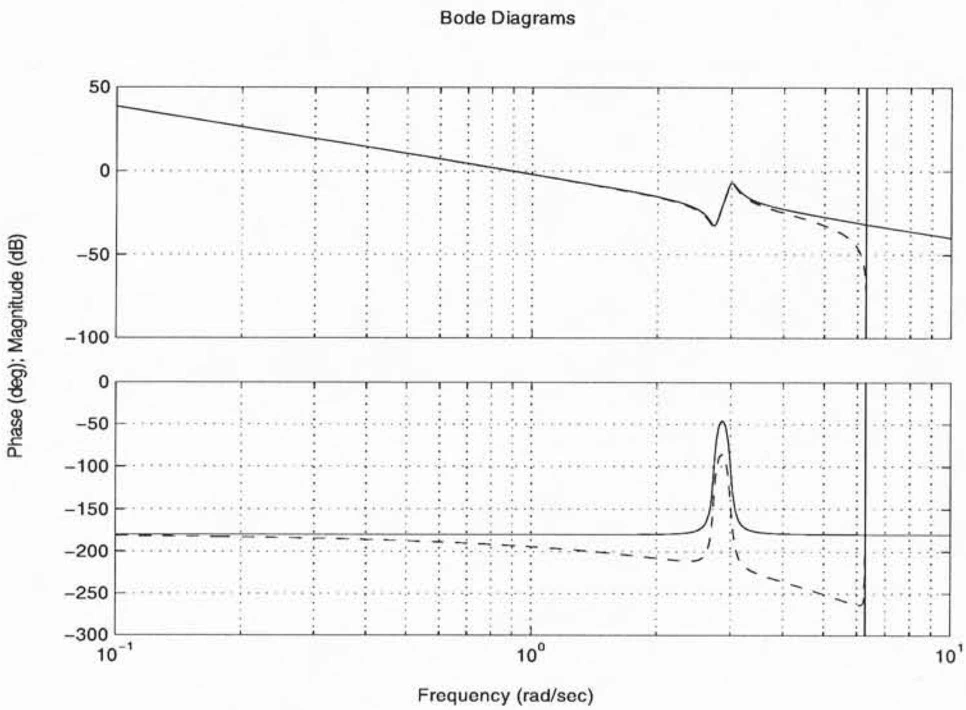


To plot the response on a wider frequency range, e.g., from 0.1 to 100 rad/sec., type

```
» bode(g,{0.1 , 100})
```

You can also discretize this system using zero-order hold and the sample time $T_s = 0.5$ second, and compare the continuous and discretized responses by:

```
» gd = c2d(g,0.5)
» bode(g,'r',gd,'b--')
```



Algorithm

For continuous-time systems, bode computes the frequency response by evaluating the transfer function $H(s)$ on the imaginary axis $s = j\omega$. Only positive frequencies ω are considered. For state-space models, the frequency response is

$$D + C(j\omega - A)^{-1}B\,, \qquad \omega \geq 0$$

When numerically safe, A is diagonalized for maximum speed. Otherwise, A is reduced to upper Hessenberg form and the linear equation $(j\omega - A)X = B$ is solved at each frequency point, taking advantage of the Hessenberg structure. The reduction to Hessenberg form provides a good compromise between efficiency and reliability. See [1] for more details on this technique.

For discrete-time systems, the frequency response is obtained by evaluating the transfer function $H(z)$ on the unit circle. To facilitate interpretation, the upper-half of the unit circle is parametrized as:

$$z = e^{j\omega T_s}, \qquad 0 \le \omega \le \omega_N = \frac{\pi}{T_s}$$

where T_s is the sample time and ω_N is called the *Nyquist frequency*. The equivalent "continuous-time frequency" ω is then used as the x-axis variable. Because $H(e^{j\omega T_s})$ is periodic with period $2\omega_N$, bode plots the response only up to the Nyquist frequency ω_N. If the sample time is unspecified, the default value $T_s = 1$ is assumed.

Diagnostics

If the system has a pole on the $j\omega$ axis (or unit circle in the discrete case) and w happens to contain this frequency point, the gain is infinite, $j\omega I - A$ is singular, and bode produces the warning message:

```
Singularity in freq. response due to jw-axis or unit circle pole.
```

See Also

`ltiview`	LTI system viewer
`nyquist`	Nyquist plot
`nichols`	Nichols plot
`sigma`	Singular value plot
`freqresp`	Frequency response computation
`evalfr`	Response at single complex frequency

References

[1] Laub, A.J., "Efficient Multivariable Frequency Response Computations," *IEEE Transactions on Automatic Control*, AC-26 (1981), pp. 407–408.

Purpose

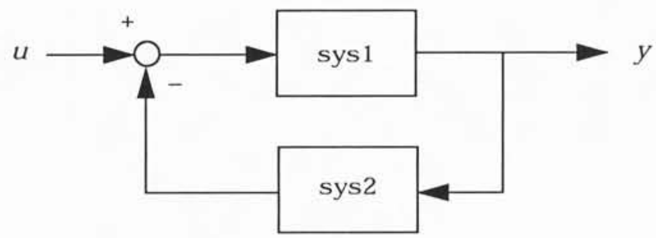
Feedback connection of two LTI models

Syntax

```
sys = feedback(sys1,sys2)
sys = feedback(sys1,sys2,sign)
sys = feedback(sys1,sys2,feedin,feedout,sign)
```

Description

`sys = feedback(sys1,sys2)` returns an LTI model sys for the negative feedback interconnection



The closed-loop model sys has u as the input vector and y as the output vector. The LTI models sys1 and sys2 must be both continuous or both discrete with identical sample times. Precedence rules are used to determine the resulting model type (see p. 2-3).

To apply positive feedback, use the syntax

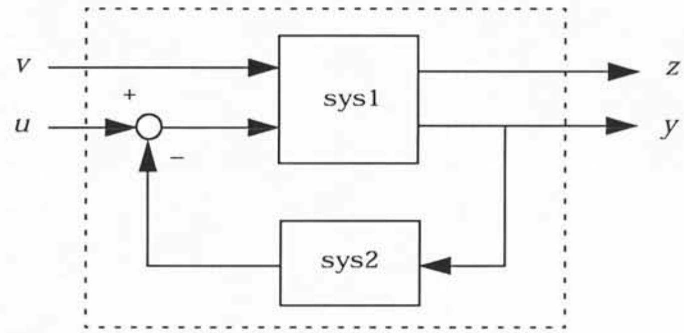
```
sys = feedback(sys1,sys2,+1)
```

By default, `feedback(sys1,sys2)` assumes negative feedback and is equivalent to `feedback(sys1,sys2,-1)`.

Finally,

```
sys = feedback(sys1,sys2,feedin,feedout)
```

computes a closed-loop model sys for the more general feedback loop:



The vector `feedin` contains indices into the input vector of `sys1` and specifies which inputs u are involved in the feedback loop. Similarly, `feedout` specifies which outputs y of `sys1` are used for feedback. The resulting LTI model `sys` has the same inputs and outputs as `sys1` (with their order preserved). As before, negative feedback is applied by default and you must use

```
sys = feedback(sys1,sys2,feedin,feedout,+1)
```

to apply positive feedback.

For more complicated feedback structures, use `append` and `connect`.

Remark

You can specify static gains as regular matrices, for example,

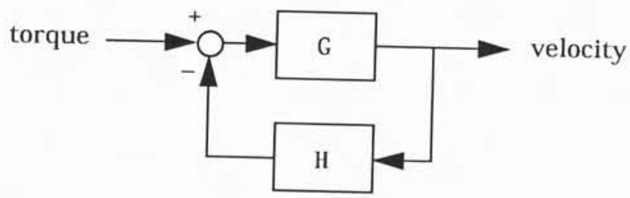
```
sys = feedback(sys1,2)
```

However, at least one of the two arguments `sys1` and `sys2` should be an LTI object. For feedback loops involving two static gains `k1` and `k2`, use the syntax

```
sys = feedback(tf(k1),k2)
```

Examples

Example 1



To connect the plant

$$G(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

with the controller

$$H(s) = \frac{5(s+2)}{s+10}$$

using negative feedback, type

```
» G = tf([2 5 1],[1 2 3],'inputname','torque',...
                          'outputname','velocity');
» H = zpk(-2,-10,5)
» Cloop = feedback(G,H)

Zero/pole/gain from input "torque" to output "velocity":
0.18182 (s+10) (s+2.281) (s+0.2192)
-----------------------------------
 (s+3.419) (s^2 + 1.763s + 1.064)
```

The result is a zero-pole-gain model as expected from the precedence rules. Note that `Cloop` inherited the input and output names from `G`.

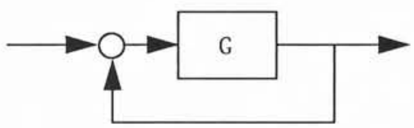
Example 2

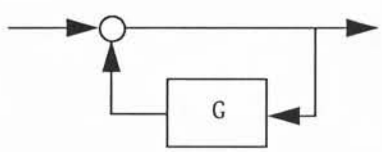
Consider a state-space plant P with five inputs and four outputs and a state-space feedback controller K with two inputs and three outputs. To connect outputs 1, 3, and 4 of the plant to the controller inputs, and the controller outputs to inputs 4 and 2 of the plant, use

```
feedin = [4 2];
feedout = [1 3 4];
Cloop = feedback(P,K,feedin,feedout)
```

Example 3

You can form the following negative-feedback loops





by

```
Cloop = feedback(G,1)     % left diagram
Cloop = feedback(1,G)     % right diagram
```

Limitations

The feedback connection should be free of algebraic loop. If D_1 and D_2 are the feedthrough matrices of sys1 and sys2, this condition is equivalent to:

- $I + D_1 D_2$ nonsingular when using negative feedback
- $I - D_1 D_2$ nonsingular when using positive feedback

See Also

star	Star product of LTI systems (LFT connection)
series	Series connection
parallel	Parallel connection
connect	Derive state-space model for block diagram interconnection
append	Append LTI systems

parallel

Purpose Parallel connection of two LTI models.

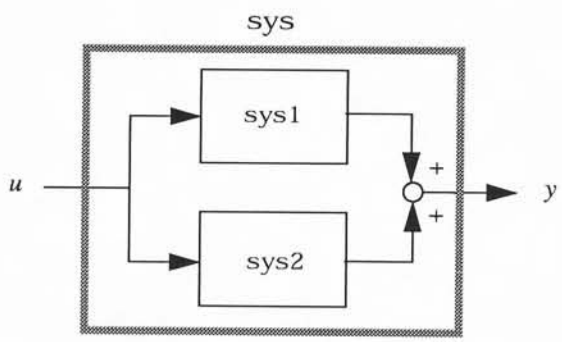
Syntax

```
sys = parallel(sys1,sys2)
sys = parallel(sys1,sys2,inp1,inp2,out1,out2)
```

Description

`parallel` connects two LTI models in parallel. This function accepts any type of LTI model. The two systems must be either both continuous or both discrete with identical sample time. Static gains are neutral and can be specified as regular matrices.

`sys = parallel(sys1,sys2)` forms the basic parallel connection shown below.

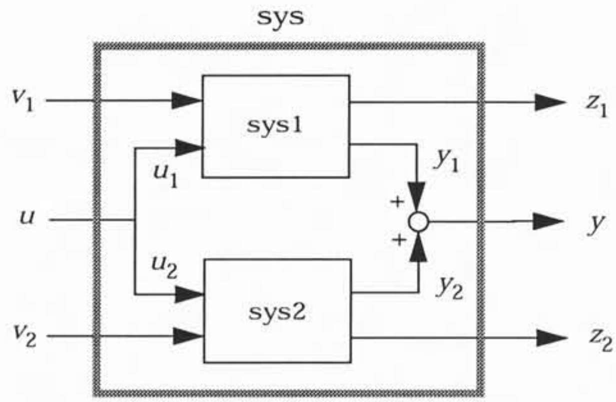


This command is equivalent to the direct addition

```
sys = sys1 + sys2
```

(see page 28 for details on LTI system addition).

`sys = parallel(sys1,sys2,inp1,inp2,out1,out2)` forms the more general parallel connection:



The index vectors `inp1` and `inp2` specify which inputs u_1 of `sys1` and which inputs u_2 of `sys2` are connected. Similarly, the index vectors `out1` and `out2` specify which outputs y_1 of `sys1` and which outputs y_2 of `sys2` are summed. The resulting model sys has $[v_1 ; u ; v_2]$ as inputs and $[z_1 ; y ; z_2]$ as outputs.

Example

See page 55 in "Design Case Studies" for an example.

See Also

`append`	Append LTI systems
`series`	Series connection
`feedback`	Feedback connection

Purpose

Series connection of two LTI models.

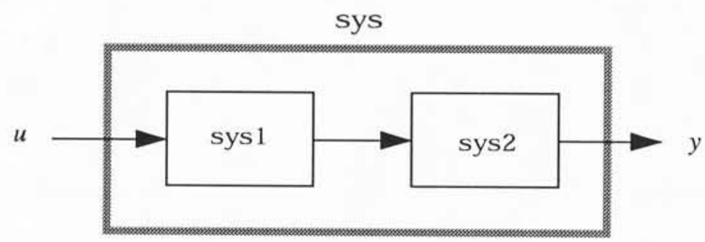
Syntax

```
sys = series(sys1,sys2)
sys = series(sys1,sys2,outputs1,inputs2)
```

Description

`series` connects two LTI models in series. This function accepts any type of LTI model. The two systems must be either both continuous or both discrete with identical sample time. Static gains are neutral and can be specified as regular matrices.

`sys = series(sys1,sys2)` forms the basic series connection shown below.

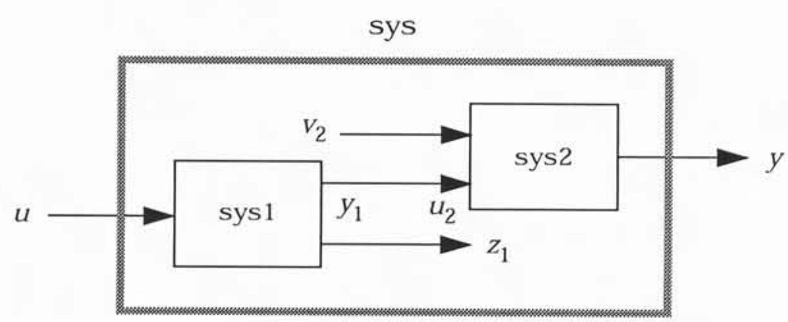


This command is equivalent to the direct multiplication

```
sys = sys2 * sys1
```

(see page 2-29 for details on LTI system multiplication).

`sys = series(sys1,sys2,outputs1,inputs2)` forms the more general series connection:



The index vectors outputs1 and inputs2 indicate which outputs y_1 of sys1 and which inputs u_2 of sys2 ought to be connected. The resulting model sys has u as input and y as output.

Example

Consider a state-space system sys1 with five inputs and four outputs and another system sys2 with two inputs and three outputs. Connect the two systems in series by connecting outputs 2 and 4 of sys1 with inputs 1 and 2 of sys2:

```
outputs1 = [2 4];
inputs2 = [1 2];
sys = series(sys1,sys2,outputs2,inputs1)
```

See Also

append	Append LTI systems
parallel	Parallel connection
feedback	Feedback connection

ET438a

Matlab Bode Function Help

BODE Bode frequency response of LTI systems.

BODE(SYS) draws the Bode plot of the LTI system SYS. The frequency range and number of points are chosen automatically.

BODE(SYS,{WMIN,WMAX}) draws the Bode plot for frequencies between WMIN and WMAX (in radian/second).

BODE(SYS,W) uses the user-supplied vector W of frequencies, in radian/second, at which the Bode response is to be evaluated. See LOGSPACE to generate logarithmically spaced frequency vectors.

BODE(SYS1,SYS2,...,W) plots the Bode response of multiple LTI systems SYS1,SYS2,... on a single plot. The frequency vector W is optional. You can also specify a color, line style, and marker for each system, as in bode(sys1,'r',sys2,'y--',sys3,'gx').

When invoked with left-hand arguments,

[MAG,PHASE,W] = BODE(SYS,...)

returns the frequency vector W and arrays MAG and PHASE of magnitudes (in dB) and phases (in degrees). No plot is drawn on the screen. If SYS has NU inputs and NY outputs and LW=length(W), MAG and PHASE are NY-by-NU-by-LW arrays and the response at the frequency W(k) is given by MAG(:,:,k) and PHASE(:,:,k).

For discrete systems with sample time Ts, BODE uses the transformation Z = exp(j*W*Ts) to map the unit circle to the real frequency axis. The frequency response is only plotted for frequencies smaller than the Nyquist frequency pi/Ts, and the default value 1 (second) is assumed when Ts is unspecified.

See also NICHOLS, NYQUIST, SIGMA, FREQRESP, LTIVIEW.