chapter12_4_1, **chapter12_4_2**, and **chapter12_4_3** Modeling in the Frequency Domain for Example 12.8

Method 1

```
% Onwubolu, G. C.
% Mechatronics: Principles & Applications
% Elsevier
%
% Mechatronics: Principles & Applications Toolbox Version 1.0
% Copyright © 2005 by Elsevier
%
% Chapter 12.4: Block diagrams
% Example 12.8, Method 1
% Solution via Series, Parallel, & Feedback Commands:     MATLAB can be
used for block diagram
% reduction. Three methods are available: (1) Solution via Series, Parallel, &
% Feedback Commands, (2) Solution via Algebraic Operations, and (3)
Solution via
% Append & Connect Commands. Let us look at each of these methods.
%
% (1) Solution via Series, Parallel, & Feedback Commands:
% The closed-loop transfer function is obtained using the following commands
% successively, where the arguments are LTI objects: series(G1,G2) for a
cascade
% connection of G1(s) and G2(s); parallel(G1,G2) for a parallel connection of
% G1(s) and G2(s); feedback(G,H,sign) for a closed-loop connection with
G(s)
% as the forward path, H(s) as the feedback, and sign is -1 for negative-
feedback
% systems or +1 for positive-feedback systems. The sign is optional for
% negative-feedback systems.
%
% (2) Solution via Algebraic Operations:
% Another approach is to use arithmetic operations successively on LTI
transfer
% functions as follows: G2*G1 for a cascade connection of G1(s) and G2(s);
G1+G2
% for a parallel connection of G1(s) and G2(s); G/(1+G*H) for a closed-loop
% negative-feedback connection with G(s) as the forward path, and H(s) as
the
% feedback; G/(1-G*H) for positive-feedback systems. When using division
we follow
% with the function minreal(sys) to cancel common terms in the numerator
% and denominator.
%
% (3) Solution via Append & Connect Commands:
% The last method, which defines the topology of the system,  may be used
effectively
```

% for complicated systems. First, the subsystems are defined. Second, the subsystems
% are appended, or gathered, into a multiple-input/multiple-output system. Think of
% this system as a single system with an input for each of the subsystems and an
% output for each of the subsystems. Next, the external inputs and outputs are
% specified. Finally, the subsystems are interconnected. Let us elaborate on each
% of these steps.
%
% The subsystems are defined by creating LTI transfer functions for each. The
% subsystems are appended using the command G = append(G1,G2,G3,G4,.....Gn), where
% the Gi are the LTI transfer funtions of the subsystems and G is the appended system.
% Each subsystem is now identified by a number based upon its position in the append
% argument. For example, G3 is 3, based on the fact that it is the third subsystem in
% the append argument (not the fact that we write it as G3).
%
% Now that we have created an appended system, we form the arguments required to
% interconnect their inputs and outputs to form our system. The first step identifies
% which subsystems have the external input signal and which subsystems have the
% external output signal. For example, we use inputs = [1 5 6] and outputs = [3 4] to
% define the external inputs to be the inputs of subsystems 1, 5 and 6 and the external
% outputs to be the outputs of subsystems 3 and 4. For single-input/single-output
% systems, these definitions use scalar quantities. Thus inputs = 5, outputs = 8 define
% the input to subsystem 5 as the external input and the output of subsystem 8 as the
% external output.
%
% At this point we tell the program how all of the subsystems are interconnected.
% We form a Q matrix that has a row for each subsystem whose input comes from another
% subsystem's output. The first column contains the subsystem's number. Subsequent
% columns contain the numbers of the subsystems from which the inputs comes. Thus,

```
% a typical row might be as follows: [3 6 -7], or subsystem 3's input is formed
from
% the sum of the output of subsystem 6 and the negative of the output of
subsystem 7.
%
% Finally, all of the interconnection arguments are used in the
% connect(G,Q,inputs,outputs) command, where all of the arguments have
been
% previously defined.
%
% Let us demonstrate the three methods for finding the total transfer function
by
% looking at the back endpapers and finding the closed-loop transfer function
of
% the pitch control loop for the UFSS with K1 = K2 = 1 (Johnson, 1980). The
last
% method using append and connect requires that all subsystems be proper
(the order
% of the numerator cannot be greater than the order of the denominator). The
pitch
% rate sensor violates this requirement. Thus, for the third method, we
perform some
% block diagram maneuvers by pushing the pitch rate sensor to the left past
the
% summing junction and combining the resulting blocks with the pitch gain
and the
% elevator actuator. These changes are reflected in the program. The student
should
% verify all computer results with hand calculations.
'Example 12.8'

'Solution via Series, Parallel, & Feedback Commands' %Dispaly label.
                              % Display label.
numg1=[-1];                          % Define numerator of G1(s).
deng1=[1];                          % Define denominator of G1(s).
numg2=[0 3];                         % Define numerator of G2(s).
deng2=[1 3];                        % Define denominator of G2(s).
numg3=-0.2*[1 0.5];                  % Define numerator of G3(s).
deng3=conv([1 1],[1 0.5 0.05]);
                              % Define denominator of G3(s).
numh1=[-1 0];                        % Define numerator of H1(s).
denh1=[0 1];                        % Define denominator of H1(s).
G1=tf(numg1,deng1);                  % Create LTI transfer function,
                              % G1(s).
G2=tf(numg2,deng2);                  % Create LTI transfer function,
                              % G2(s).
G3=tf(numg3,deng3);                  % Create LTI transfer function,
                              % G3(s).
H1=tf(numh1,denh1);                  % Create LTI transfer function,
                              % H1(s).
```

```
G4=series(G2,G3);                 % Calculate product of elevator and
                                  % vehicle dynamics.
G5=feedback(G4,H1);               % Calculate closed-loop transfer
                                  % function of inner loop.
Ge=series(G1,G5);                 % Multiply inner-loop transfer
                                  % function and pitch gain.
'T(s) via Series, Parallel, & Feedback Commands'
                                  % Display label.
T=feedback(Ge,1)                  % Find closed-loop transfer function.
Pause
```

Method 2

```
% Onwubolu, G. C.
% Mechatronics: Principles & Applications
% Elsevier
%
% Mechatronics: Principles & Applications Toolbox Version 1.0
% Copyright © 2005 by Elsevier
%
% Chapter 12.4: Block diagrams
% Example 12.8, Method 2
% Solution via Algebraic Operations:     MATLAB can be used for block
diagram
% reduction. Three methods are available: (1) Solution via Series, Parallel, &
% Feedback Commands, (2) Solution via Algebraic Operations, and (3)
Solution via
% Append & Connect Commands. Let us look at each of these methods.
%
% (1) Solution via Series, Parallel, & Feedback Commands:
% The closed-loop transfer function is obtained using the following commands
% successively, where the arguments are LTI objects: series(G1,G2) for a
cascade
% connection of G1(s) and G2(s); parallel(G1,G2) for a parallel connection of
% G1(s) and G2(s); feedback(G,H,sign) for a closed-loop connection with
G(s)
% as the forward path, H(s) as the feedback, and sign is -1 for negative-
feedback
% systems or +1 for positive-feedback systems. The sign is optional for
% negative-feedback systems.
%
% (2) Solution via Algebraic Operations:
% Another approach is to use arithmetic operations successively on LTI
transfer
% functions as follows: G2*G1 for a cascade connection of G1(s) and G2(s);
G1+G2
% for a parallel connection of G1(s) and G2(s); G/(1+G*H) for a closed-loop
% negative-feedback connection with G(s) as the forward path, and H(s) as
the
```

% feedback; G/(1-G*H) for positive-feedback systems. When using division we follow
% with the function minreal(sys) to cancel common terms in the numerator
% and denominator.
%
% (3) Solution via Append & Connect Commands:
% The last method, which defines the topology of the system, may be used effectively
% for complicated systems. First, the subsystems are defined. Second, the subsystems
% are appended, or gathered, into a multiple-input/multiple-output system. Think of
% this system as a single system with an input for each of the subsystems and an
% output for each of the subsystems. Next, the external inputs and outputs are
% specified. Finally, the subsystems are interconnected. Let us elaborate on each
% of these steps.
%
% The subsystems are defined by creating LTI transfer functions for each. The
% subsystems are appended using the command G = append(G1,G2,G3,G4,.....Gn), where
% the Gi are the LTI transfer funtions of the subsystems and G is the appended system.
% Each subsystem is now identified by a number based upon its position in the append
% argument. For example, G3 is 3, based on the fact that it is the third subsystem in
% the append argument (not the fact that we write it as G3).
%
% Now that we have created an appended system, we form the arguments required to
% interconnect their inputs and outputs to form our system. The first step identifies
% which subsystems have the external input signal and which subsystems have the
% external output signal. For example, we use inputs = [1 5 6] and outputs = [3 4] to
% define the external inputs to be the inputs of subsystems 1, 5 and 6 and the external
% outputs to be the outputs of subsystems 3 and 4. For single-input/single-output
% systems, these definitions use scalar quantities. Thus inputs = 5, outputs = 8 define
% the input to subsystem 5 as the external input and the output of subsystem 8 as the
% external output.
%

```
% At this point we tell the program how all of the subsystems are
interconnected.
% We form a Q matrix that has a row for each subsystem whose input comes
from another
% subsystem's output. The first column contains the subsystem's number.
Subsequent
% columns contain the numbers of the subsystems from which the inputs
comes. Thus,
% a typical row might be as follows: [3 6 -7], or subsystem 3's input is formed
from
% the sum of the output of subsystem 6 and the negative of the output of
subsystem 7.
%
% Finally, all of the interconnection arguments are used in the
% connect(G,Q,inputs,outputs) command, where all of the arguments have
been
% previously defined.
%
% Let us demonstrate the three methods for finding the total transfer function
by
% looking at the back endpapers and finding the closed-loop transfer function
of
% the pitch control loop for the UFSS with K1 = K2 = 1 (Johnson, 1980). The
last
% method using append and connect requires that all subsystems be proper
(the order
% of the numerator cannot be greater than the order of the denominator). The
pitch
% rate sensor violates this requirement. Thus, for the third method, we
perform some
% block diagram maneuvers by pushing the pitch rate sensor to the left past
the
% summing junction and combining the resulting blocks with the pitch gain
and the
% elevator actuator. These changes are reflected in the program. The student
should
% verify all computer results with hand calculations.
'Example 12.8'

'Solution via Algebraic Operations'
                                % Display label.
numg1=[-1];                     % Define numerator of G1(s).
deng1=[1];                      % Define denominator of G1(s).
numg2=[0 3];                    % Define numerator of G2(s).
deng2=[1 3];                    % Define denominator of G2(s).
numg3=-0.2*[1 0.5];             % Define numerator of G3(s).
deng3=conv([1 1],[1 0.5 0.05]);
                                % Define denominator of G3(s).
numh1=[-1 0];                   % Define numerator of H1(s).
denh1=[0 1];                    % Define denominator of H1(s).
```

```
G1=tf(numg1,deng1);              % Create LTI transfer function,
                                 % G1(s).
G2=tf(numg2,deng2);              % Create LTI transfer function,
                                 % G2(s).
G3=tf(numg3,deng3);              % Create LTI transfer function,
                                 % G3(s).
H1=tf(numh1,denh1);              % Create LTI transfer function,
                                 % H1(s).
G4=G3*G2;                        % Calculate product of elevator and
                                 % vehicle dynamics.
G5=G4/(1+G4*H1);                 % Calculate closed-loop transfer
                                 % function of inner loop.
G5=minreal(G5);                  % Cancel common terms.
Ge=G5*G1                         % Multiply inner-loop transfer
                                 % functions.
Pause
```

Method 3

```
% Onwubolu, G. C.
% Mechatronics: Principles & Applications
% Elsevier
%
% Mechatronics: Principles & Applications Toolbox Version 1.0
% Copyright © 2005 by Elsevier
%
% Chapter 12.4: Block diagrams
% Example 12.8, Method 3
% Solution via Append & Connect Commands:     MATLAB can be used for
block diagram
% reduction. Three methods are available: (1) Solution via Series, Parallel, &
% Feedback Commands, (2) Solution via Algebraic Operations, and (3)
Solution via
% Append & Connect Commands. Let us look at each of these methods.
%
% (1) Solution via Series, Parallel, & Feedback Commands:
% The closed-loop transfer function is obtained using the following commands
% successively, where the arguments are LTI objects: series(G1,G2) for a
cascade
% connection of G1(s) and G2(s); parallel(G1,G2) for a parallel connection of
% G1(s) and G2(s); feedback(G,H,sign) for a closed-loop connection with
G(s)
% as the forward path, H(s) as the feedback, and sign is -1 for negative-
feedback
% systems or +1 for positive-feedback systems. The sign is optional for
% negative-feedback systems.
%
% (2) Solution via Algebraic Operations:
% Another approach is to use arithmetic operations successively on LTI
transfer
```

% functions as follows: G2*G1 for a cascade connection of G1(s) and G2(s); G1+G2
% for a parallel connection of G1(s) and G2(s); G/(1+G*H) for a closed-loop
% negative-feedback connection with G(s) as the forward path, and H(s) as the
% feedback; G/(1-G*H) for positive-feedback systems. When using division we follow
% with the function minreal(sys) to cancel common terms in the numerator
% and denominator.
%
% (3) Solution via Append & Connect Commands:
% The last method, which defines the topology of the system, may be used effectively
% for complicated systems. First, the subsystems are defined. Second, the subsystems
% are appended, or gathered, into a multiple-input/multiple-output system. Think of
% this system as a single system with an input for each of the subsystems and an
% output for each of the subsystems. Next, the external inputs and outputs are
% specified. Finally, the subsystems are interconnected. Let us elaborate on each
% of these steps.
%
% The subsystems are defined by creating LTI transfer functions for each. The
% subsystems are appended using the command G = append(G1,G2,G3,G4,.....Gn), where
% the Gi are the LTI transfer funtions of the subsystems and G is the appended system.
% Each subsystem is now identified by a number based upon its position in the append
% argument. For example, G3 is 3, based on the fact that it is the third subsystem in
% the append argument (not the fact that we write it as G3).
%
% Now that we have created an appended system, we form the arguments required to
% interconnect their inputs and outputs to form our system. The first step identifies
% which subsystems have the external input signal and which subsystems have the
% external output signal. For example, we use inputs = [1 5 6] and outputs = [3 4] to
% define the external inputs to be the inputs of subsystems 1, 5 and 6 and the external
% outputs to be the outputs of subsystems 3 and 4. For single-input/single-output

```
% systems, these definitions use scalar quantities. Thus inputs = 5, outputs =
8 define
% the input to subsystem 5 as the external input and the output of subsystem
8 as the
% external output.
%
% At this point we tell the program how all of the subsystems are
interconnected.
% We form a Q matrix that has a row for each subsystem whose input comes
from another
% subsystem's output. The first column contains the subsystem's number.
Subsequent
% columns contain the numbers of the subsystems from which the inputs
comes. Thus,
% a typical row might be as follows: [3 6 -7], or subsystem 3's input is formed
from
% the sum of the output of subsystem 6 and the negative of the output of
subsystem 7.
%
% Finally, all of the interconnection arguments are used in the
% connect(G,Q,inputs,outputs) command, where all of the arguments have
been
% previously defined.
%


'Solution via Append & Connect Commands'
                              % Display label.
'G1(s) = (-1)*(1/(-s)) = 1/s'      % Display label.
numg1=[1];                         % Define numerator of G1(s).
deng1=[1 0];                       % Define denominator of G1(s).
G1=tf(numg1,deng1)                      % Create LTI transfer function,
                              % G1(s) = pitch gain*(1/pitch rate sensor).
'G2(s) = (-s)*(3/(s+3)'            % Display label.
numg2=[-3 0];                       % Define numerator of G2(s).
deng2=[1 3];                       % Define denominator of G2(s).
G2=tf(numg2,deng2)                      % Create LTI transfer function,
                              % G2(s) = pitch rate sensor* vehicle dynamics.
'G3(s) = -0.2(s+0.5)/((s+1)(s^2+0.5s+0.05))'
                              % Display label.
numg3=-0.2*[1 0.5];                  % Define numerator of G3(s).
deng3=conv([1 1],[1 0.5 0.05]);
                              % Define denominator of G3(s).
G3=tf(numg3,deng3)                      % Create LTI transfer function,
                              % G3(s) = vehicle dynamics.
System=append(G1,G2,G3);           % Gather all subsystems
input=1;                         % Input is at first subsystem, G1(s).
output=3;                         % Output is output of third subsystem, G3(s).
Q=[1 -3 0                         % Subsystem 1, G1(s), gets its input from the
                              % negative of the output of subsystem 3, G3(s).
```

```
2 1 -3                          % Subsystem 2, G2(s), gets its input from subsystem
                                % 1, G1(s), and the negative of the output of
                                % subsystem 3, G3(s).
3 2 0];                         % Subsystem 3, G3(s), gets its input from subsystem
                                % 2, G2(s).
T=connect(System,Q,input,output);   % Connect the subsystems.
'T(s) via Append & Connect Commands'% Display label.
T=tf(T);                        % Create LTI closed-loop transfer function,
T=minreal(T)                    % Cancel common terms.
pause
```