2

Preliminary Study of MATLAB Analysis of Dynamic Systems

2–1 PARTIAL-FRACTION EXPANSION WITH MATLAB

For transfer functions with denominators involving higher-order polynomials, hand computation of partial-fraction expansion may be quite time consuming. In such a case, the use of MATLAB is recommended. MATLAB has a command to obtain the partial-fraction expansion of $B(s)/A(s)$. It also has a command to obtain the zeros and poles of $B(s)/A(s)$.

First we present the MATLAB approach to obtaining the partial-fraction expansion of $B(s)/A(s)$. Then we discuss the MATLAB approach to obtaining the zeros and poles of $B(s)/A(s)$.

Partial-fraction expansion with MATLAB. Consider the function

$$\frac{B(s)}{A(s)} = \frac{\text{num}}{\text{den}} = \frac{b_0 s^n + b_1 s^{n-1} + \cdots + b_n}{s^n + a_1 s^{n-1} + \cdots + a_n}$$

where some of a_i and b_j may be zero. In MATLAB, row vectors num and den specify the coefficients of the numerator and denominator of the transfer function. That is,

```
num = [b0 b1 ... bn]
den = [1 a1 ... an]
```

The command

```
[r,p,k] = residue(num,den)
```

finds the residues (r), poles (p), and direct terms (k) of a partial-fraction expansion of the ratio of two polynomials $B(s)$ and $A(s)$.

The partial-fraction expansion of $B(s)/A(s)$ is given by

$$\frac{B(s)}{A(s)} = \frac{r(1)}{s - p(1)} + \frac{r(2)}{s - p(2)} + \cdots + \frac{r(n)}{s - p(n)} + k(s) \tag{2-1}$$

EXAMPLE 2-1 Consider the transfer function

$$\frac{B(s)}{A(s)} = \frac{2s^3 + 5s^2 + 3s + 6}{s^3 + 6s^2 + 11s + 6}$$

For this function,

```
num = [2   5   3   6]
den = [1   6   11   6]
```

The command

```
[r,p,k] = residue(num,den)
```

gives the following result:

```
>> num = [2    5    3    6];
>> den = [1    6    11    6];
>> [r, p, k] = residue (num,den)
r =

   -6.0000
   -4.0000
    3.0000

p =

   -3.0000
   -2.0000
   -1.0000
k =

    2
```

(Note that the residues are returned in column vector r, the pole locations in column vector p, and the direct term in row vector k.) This is the MATLAB representation of the following partial-fraction expansion of $B(s)/A(s)$:

$$\begin{aligned}\frac{B(s)}{A(s)} &= \frac{2s^3 + 5s^2 + 3s + 6}{s^3 + 6s^2 + 11s + 6} \\ &= \frac{-6}{s+3} + \frac{-4}{s+2} + \frac{3}{s+1} + 2\end{aligned}$$

The residue command can also be used to form the polynomials (numerator and denominator) from the partial-fraction expansion. That is, the command

```
[num,den] = residue(r,p,k)
```

where r, p, and k are as given in the previous MATLAB output, converts the partial-fraction expansion back to the polynomial ratio $B(s)/A(s)$, as follows:

```
>> r = [-6   -4   3];
>> p = [-3   -2   -1];
>> k = 2;
>> [num,den] = residue(r,p,k)

num =

     2     5     3     6


den =

     1     6    11     6
```

Or we may enter the following MATLAB program to get $B(s)/A(s)$ = num/den:

```
>> [num,den] = residue(r,p,k);
>> printsys(num,den,'s')
num/den =

    2s^3 + 5s^2 + 3s + 6
    --------------------
    s^3 + 6s^2 + 11s + 6
```

The command

```
printsys(num,den,'s')
```

prints num/den in terms of the ratio of polynomials in s.

Note that if $p(j) = p(j+1) = \cdots = p(j+m-1)$ (that is, $p_j = p_{j+1} = \cdots = p_{j+m-1}$), then the pole $p(j)$ is a pole of multiplicity m. In such a case, the expansion includes terms of the form

$$\frac{r(j)}{s-p(j)} + \frac{r(j+1)}{[s-p(j)]^2} + \cdots + \frac{r(j+m-1)}{[s-p(j)]^m}$$

(For details, see Example 2–2.)

EXAMPLE 2–2 Expand

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

into partial-fractions with MATLAB.

For this function, we have

```
num = [0   1   2   3]
den = [1   3   3   1]
```

Note that for the numerator either [0 1 2 3] or [1 2 3] may be used. The command

```
[r,p,k] = residue(num,den)
```

gives the following result:

```
>> num = [1   2   3];
>> den = [1   3   3   1];
>> [r,p,k] = residue(num,den)
r =

    1.0000
    0.0000
    2.0000

p =

   -1.0000   s
   -1.0000   s2
   -1.0000   s3

k =

    []
```

This output is the MATLAB representation of the partial-fraction expansion of

$$\frac{B(s)}{A(s)} = \frac{1}{s+1} + \frac{0}{(s+1)^2} + \frac{2}{(s+1)^3}$$

Note that the direct term k is zero.

To obtain the original function $B(s)/A(s)$ from r, p, and k, enter the following program into the computer:

```
>> [num,den] = residue(r,p,k);
>> printsys(num,den,'s')
```

The computer will then show

```
num/den = s^2 + 2s + 3
          ---------------------
          s^3 + 3s^2 + 3s + 1
```

Finding zeros and poles of $B(s)/A(s)$ with MATLAB. The MATLAB command

[z,p,K] = tf2zp(num,den)

obtains the zeros, poles, and gain K of $B(s)/A(s)$.

tf2zp

Consider the system defined by

$$\frac{B(s)}{A(s)} = \frac{4s^2 + 16s + 12}{s^4 + 12s^3 + 44s^2 + 48s}$$

To obtain the zeros (z), poles (p), and gain (K), enter the following MATLAB program into the computer:

```
>> num = [4   16   12];
>> den = [1   12   44   48   0];
>> [z,p,K] = tf2zp(num,den)
```

The computer will then produce the following output on the screen:

```
z =

   -3
   -1

p =

    0
   -6.0000
   -4.0000
   -2.0000

K =

   4
```

The zeros are at $s = -3$ and -1. The poles are at $s = 0, -6, -4,$ and -2. The gain K is 4.

If the zeros, poles, and gain K are given, then the following MATLAB program will yield the original num/den:

```
>> z = [-1; -3];
>> p = [0; -2; -4; -6];
>> K = 4;
>> [num,den] = zp2tf(z,p,K);
>> printsys(num,den,'s')

num/den =
        4s^2 + 16s + 12
   ---------------------------
   s^4 + 12s^3 + 44s^2 + 48s
```

zp2tf

EXAMPLE 2-3 Obtain the zeros (z), poles (p), and gain (K) of the system

$$\frac{B(s)}{A(s)} = \frac{5s^3 + 30s^2 + 55s + 30}{s^3 + 9s^2 + 33s + 65}$$

MATLAB Program 2–1 gives the desired result.

MATLAB Program 2–1

```
>> num = [5   30   55   30];
>> den = [1   9   33   65];
>> [z,p,K] = tf2zp(num,den)

z =

   -3.0000
   -2.0000
   -1.0000

p =

   -5.0000
   -2.0000 + 3.0000i
   -2.0000 - 3.0000i

K =

     5
```

Next, given the zeros (z), poles (p), and gain (K), as obtained in MATLAB Program 2–1, determine the numerator and denominator of $B(s)/A(s)$ and also the transfer function $B(s)/A(s)$ = num/den. MATLAB Program 2–2 yields the desired result.

MATLAB Program 2–2

```
>> z = [-1; -2; -3];
>> p = [-2+j*3;-2-j*3;-5];
>> K = 5;
>> [num,den] = zp2tf(z,p,K)
num =

     5    30    55    30
den =

     1     9    33    65
>> printsys(num,den,'s')
num/den =

   5 s^3 + 30 s^2 + 55 s + 30
   --------------------------
    s^3 + 9 s^2 + 33 s + 65
```

EXAMPLE 2-4 Consider the mechanical system shown in Figure 2-1. The system is at rest initially. The displacements x and y are measured from their respective equilibrium positions. Assuming that $p(t)$ is a step force input and the displacement $x(t)$ is the output, obtain the transfer function of the system. Then, assuming that $m = 0.1$ kg, $b_2 = 0.4$ N-s/m, $k_1 = 6$ N/m, $k_2 = 4$ N/m, and $p(t)$ is a step force of magnitude 10 N, obtain an analytical solution $x(t)$.

The equations of motion for the system are

$$m\ddot{x} + k_1 x + k_2(x - y) = p$$
$$k_2(x - y) = b_2\dot{y}$$

Laplace transforming these two equations, assuming zero initial conditions, we obtain

$$(ms^2 + k_1 + k_2)X(s) = k_2 Y(s) + P(s) \tag{2-2}$$
$$k_2 X(s) = (k_2 + b_2 s)Y(s) \tag{2-3}$$

Solving Equation (2-3) for $Y(s)$ and substituting the result into Equation (2-2), we get

$$(ms^2 + k_1 + k_2)X(s) = \frac{k_2^2}{k_2 + b_2 s}X(s) + P(s)$$

or

$$[(ms^2 + k_1 + k_2)(k_2 + b_2 s) - k_2^2]X(s) = (k_2 + b_2 s)P(s)$$

from which we obtain the transfer function

$$\frac{X(s)}{P(s)} = \frac{b_2 s + k_2}{mb_2 s^3 + mk_2 s^2 + (k_1 + k_2)b_2 s + k_1 k_2} \tag{2-4}$$

Substituting the given numerical values for m, k_1, k_2, and b_2 into Equation (2-4), we have

$$\frac{X(s)}{P(s)} = \frac{0.4s + 4}{0.04s^3 + 0.4s^2 + 4s + 24}$$
$$= \frac{10s + 100}{s^3 + 10s^2 + 100s + 600} \tag{2-5}$$

Since $P(s)$ is a step force of magnitude 10 N,

$$P(s) = \frac{10}{s}$$

$m\ddot{x} + k_1(x) + k_2(x-y) = P(t)$

$k_2(x-y) = b_2\dot{y}$

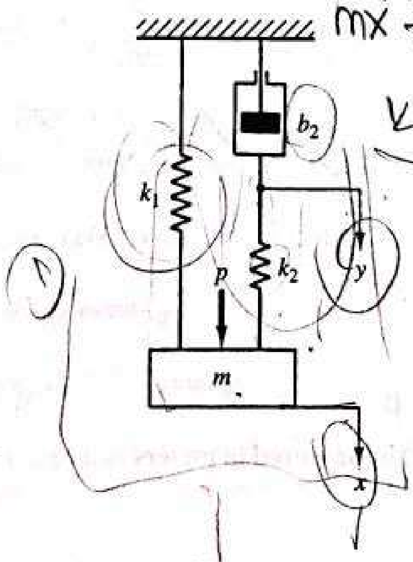


Figure 2-1
Mechanical system.

Then, from Equation (2–5),

$$X(s) = \frac{10s + 100}{s^3 + 10s^2 + 100s + 600}\frac{10}{s}$$

To find an analytical solution, we need to expand $X(s)$ into partial fractions. For this purpose, we may use the following MATLAB program to find the residues, poles, and direct term:

```
>> num = [100   1000];
>> den = [1   10   100   600   0];
>> [r,p,k] = residue(num, den)

r =
   -0.6845 + 0.2233i
   -0.6845 - 0.2233i
   -0.2977
    1.6667

p =
   -1.2898 + 8.8991i
   -1.2898 - 8.8991i
   -7.4204
        0

k =
        []
```

On the basis of the MATLAB output, $X(s)$ can be written as

$$X(s) = \frac{-0.6845 + j0.2233}{s + 1.2898 - j8.8991} + \frac{-0.6845 - j0.2233}{s + 1.2898 + j8.8991}$$
$$+ \frac{-0.2977}{s + 7.4204} + \frac{1.6667}{s}$$
$$= \frac{-1.3690(s + 1.2898) - 3.9743}{(s + 1.2898)^2 + 8.8991^2} - \frac{0.2977}{s + 7.4204} + \frac{1.6667}{s}$$

The inverse Laplace transform of $X(s)$ gives

$$x(t) = -1.3690e^{-1.2898t}\cos(8.8991t)$$
$$-0.4466e^{-1.2898t}\sin(8.8991t) - 0.2977e^{-7.4204t} + 1.6667$$

where $x(t)$ is measured in meters and time t in seconds. This is the analytical solution to the problem.

EXAMPLE 2-5 Obtain the inverse Laplace transform of

$$F(s) = \frac{s^5 + 8s^4 + 23s^3 + 35s^2 + 28s + 3}{s^3 + 6s^2 + 8s}$$

[Use MATLAB to find the partial-fraction expansion of $F(s)$.]

The following MATLAB program will produce the partial-fraction expansion of $F(s)$:

```
>> num = [1   8   23   35   28   3];
>> den = [1   6   8   0];
>> [r,p,k] = residue(num,den)
r =

   0.3750
   0.2500
   0.3750

p =

   -4
   -2
    0

k =

   1   2   3
```

Note that k = [1 2 3] means that $F(s)$ involves $s^2 + 2s + 3$ as follows:

$$F(s) = s^2 + 2s + 3 + \frac{0.375}{s+4} + \frac{0.25}{s+2} + \frac{0.375}{s}$$

Hence, the inverse Laplace transform of $F(s)$ is given by

$$f(t) = \frac{d^2}{dt^2}\delta(t) + 2\frac{d}{dt}\delta(t) + 3\delta(t) + 0.375e^{-4t} + 0.25e^{-2t} + 0.375, \qquad \text{for } t > 0-$$

EXAMPLE 2-6 Given the zero(s), pole(s), and gain K of $B(s)/A(s)$, use MATLAB to obtain the function $B(s)/A(s)$. Consider the following three cases:

1. There is no zero. Poles are at $-1 + 2j$ and $-1 - 2j$. $K = 10$.
2. A zero is at 0. Poles are at $-1 + 2j$ and $-1 - 2j$. $K = 10$.
3. A zero is at -1. Poles are at -2, -4 and -8. $K = 12$.

MATLAB programs to obtain $B(s)/A(s)$ = num/den for the three cases are as follows:

```
>> z = [];
>> p = [-1+2*j;-1-2*j];
>> K = 10;
>> [num,den] = zp2tf(z,p,K);
>> printsys(num,den)

num/den =
       10
  --------------
  s^2 + 2s + 5
```

```
>> z = [0];
>> p = [-1+2*j; -1-2*j];
>> K = 10;
>> [num,den] = zp2tf(z,p,K);
>> printsys(num,den)

num/den =
       10s
  --------------
  s^2 + 2s + 5
```

```
>> z = [-1];
>> p = [-2; -4; -8];
>> K = 12;
>> [num,den] = zp2tf(z,p,K);
>> printsys(num,den)

num/den =
             12s + 12
  ---------------------------
  s^3 + 14s^2 + 56s + 64
```

EXAMPLE 2-7 Solve the differential equation

$$\ddot{x} + 2\dot{x} + 10x = t^2, \qquad x(0) = 0, \qquad \dot{x}(0) = 0$$

Since the initial conditions are zeros, the Laplace transform of the equation becomes

$$s^2X(s) + 2sX(s) + 10X(s) = \frac{2}{s^3}$$

Hence,

$$X(s) = \frac{2}{s^3(s^2 + 2s + 10)}$$

We need to find the partial-fraction expansion of $X(s)$. Since the denominator involves a triple pole, it is simpler to use MATLAB to obtain the partial-fraction expansion. The following MATLAB program may be used:

```
>> num = [2];
>> den = [1  2  10  0  0  0];
>> [r,p,k] = residue(num,den)
r =

   0.0060 - 0.0087i
   0.0060 + 0.0087i
  -0.0120
  -0.0400
   0.2000

p =

  -1.0000 + 3.0000i
  -1.0000 - 3.0000i
   0
   0
   0

k =

  []
```

From the MATLAB output, we find that

$$X(s) = \frac{0.006 - 0.0087j}{s + 1 - 3j} + \frac{0.006 + 0.0087j}{s + 1 + 3j} + \frac{-0.012}{s} + \frac{-0.04}{s^2} + \frac{0.2}{s^3}$$

Combining the first two terms on the right-hand side of the equation, we get

$$X(s) = \frac{0.012(s + 1) + 0.0522}{(s + 1)^2 + 3^2} - \frac{0.012}{s} - \frac{0.04}{s^2} + \frac{0.2}{s^3}$$

The inverse Laplace transform of $X(s)$ gives

$$x(t) = 0.012e^{-t}\cos 3t + 0.0174e^{-t}\sin 3t - 0.012 - 0.04t + 0.1t^2, \qquad \text{for } t \geq 0$$

Note that if only the roots of a polynomial such as

$$d(s) = s^3 + 2s^2 + 3s + 4$$

are desired, we may use the command

r = roots(d)

(See MATLAB Program 2–3.) The command poly(r) produces the original polynomial such that

d = poly(r)

(See also MATLAB Program 2–3.)

MATLAB Program 2–3

```
>> d = [1   2   3   4];
>> r = roots(d)
r =

   -1.6506
   -0.1747 + 1.5469i
   -0.1747 - 1.5469i

>> d = poly(r)
d =

    1.0000    2.0000    3.0000    4.0000
```

2–2 TRANSFORMATION OF MATHEMATICAL MODELS OF DYNAMIC SYSTEMS

MATLAB has useful commands to transform a mathematical model of a linear time-invariant system to another model. The following are linear system transformations that are useful for solving control engineering problems:

Transfer-function to state-space conversion (tf2ss)
State-space to transfer-function conversion (ss2tf)
State-space to zero-pole conversion (ss2zp)
Zero-pole to state-space conversion (zp2ss)
Transfer-function to zero-pole conversion (tf2zp)
Zero-pole to transfer-function conversion (zp2tf)
Continuous-time to discrete-time conversion (c2d)

We begin our discussion with the transformation from the transfer function to state space.

Transfer function to state space. The command

[A,B,C,D] = tf2ss(num,den)

converts a system from the transfer function form

$$\frac{Y(s)}{U(s)} = \frac{\text{num}}{\text{den}} = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + D$$

to the state-space form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u$$
$$y = \mathbf{C}\mathbf{x} + Du$$

It is important to note that the state-space representation of any system is not unique: There are many (indeed, infinitely many) state-space representations of the same system. The MATLAB command gives one possible such representation.

Consider the transfer-function system

$$\frac{Y(s)}{U(s)} = \frac{10s + 10}{s^3 + 6s^2 + 5s + 10} \tag{2–6}$$

There are many (again, infinitely many) possible state-space representations of this system. One possible such representation is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -10 & -5 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 10 \\ -50 \end{bmatrix} u$$

$$y = [1 \quad 0 \quad 0] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + [0]u$$

Another possible state-space representation (among infinitely many alternatives) is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -6 & -5 & -10 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u \tag{2–7}$$

$$y = [0 \quad 10 \quad 10] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + [0]u \tag{2–8}$$

MATLAB Program 2–4 transforms the transfer function given by Equation (2–6) into the state-space representation given by Equations (2–7) and (2–8).

MATLAB Program 2–4

```
>> num = [10   10];
>> den = [1   6   5   10];
>> [A,B,C,D] = tf2ss(num,den)
A =
    -6   -5   -10
     1    0     0
     0    1     0
B =
     1
     0
     0
C =
     0   10   10
D =
     0
```

EXAMPLE 2–8 Consider the transfer function system

$$\frac{Y(s)}{U(s)} = \frac{25.04s + 5.008}{s^3 + 5.03247s^2 + 25.1026s + 5.008}$$

Obtain a state-space representation of this system with MATLAB.

MATLAB command

[A,B,C,D] = tf2ss(num,den)

will produce a state-space representation of the system. (See MATLAB Program 2–5.)

MATLAB Program 2–5

```
>> num = [25.04   5.008];
>> den = [1   5.03247   25.1026   5.008];
>> [A,B,C,D] = tf2ss(num,den)
A =

   -5.0325   -25.1026   -5.0080
    1.0000          0         0
         0     1.0000         0

B =

     1
     0
     0

C =

     0   25.0400   5.0080

D =

     0
```

This is the MATLAB representation of the following state-space equations:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -5.0325 & -25.1026 & -5.008 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u$$

$$y = [0 \quad 25.04 \quad 5.008] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + [0]u$$

Transformation from state space to transfer function. To obtain the transfer function from state-space equations, use the command

[num,den] = ss2tf(A,B,C,D,iu)

iu must be specified for systems with more than one input. For example, if the system has three inputs ($u1, u2, u3$), then iu must be either 1, 2, or 3, where 1 implies $u1$, 2 implies $u2$, and 3 implies $u3$.

If the system has only one input, then either

```
[num,den] = ss2tf(A,B,C,D)
```

or

```
[num,den] = ss2tf(A,B,C,D,1)
```

may be used. (See Example 2–9 and MATLAB Program 2–6.)

EXAMPLE 2–9 Obtain the transfer function of the system defined by the following state-space equations:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -5.008 & -25.1026 & -5.03247 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 25.04 \\ -121.005 \end{bmatrix} u$$

$$y = [1 \quad 0 \quad 0] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

MATLAB Program 2–6 will produce the transfer function of the given system, namely,

$$\frac{Y(s)}{U(s)} = \frac{25.04s + 5.008}{s^3 + 5.0325s^2 + 25.1026s + 5.008}$$

MATLAB Program 2–6

```
>> A = [0   1   0;0   0   1;-5.008   -25.1026   -5.03247];
>> B = [0;25.04; -121.005];
>> C = [1   0   0];
>> D = [0];
>> [num,den] = ss2tf(A,B,C,D)

num =

         0   -0.0000   25.0400   5.0080

den =

    1.0000   5.0325   25.1026   5.0080

>> % *****The same result can be obtained by entering the following command *****

>> [num,den] = ss2tf(A,B,C,D,1)

num =

         0   -0.0000   25.0400   5.0080

den =

    1.0000   5.0325   25.1026   5.0080
```

If the system involves one output but more than one input, use the command

[num,den] = ss2tf(A,B,C,D,iu)

This command converts the state-space system

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$
$$y = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

to the transfer function system

$$\frac{Y(s)}{U_i(s)} = i\text{th element of } [\mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}]$$

Note that the scalar iu is an index into the inputs of the system and specifies which input is to be used for the response.

EXAMPLE 2–10 Consider the system

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$y = [1 \quad 0]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0 \quad 0]\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

which has two inputs and one output.

Two transfer functions may be obtained for this system. One relates the output y and input u_1, and the other relates the output y and input u_2. (When considering input u_1, we assume that input u_2 is zero, and vice versa.) MATLAB Program 2–7 produces the transfer functions in question.

MATLAB Program 2–7

```
>> A = [0   1;-2   -3];
>> B = [1   0;0   1];
>> C = [1   0];
>> D = [0   0];
>> [num,den] = ss2tf(A, B, C, D, 1)

num =

        0      1.0000      3.0000

den =

        1      3      2

>> [num,den] = ss2tf(A, B, C, D, 2)

num =

        0      0.0000      1.0000

den =

        1      3      2
```

From the MATLAB output, we have

$$\frac{Y(s)}{U_1(s)} = \frac{s+3}{s^2+3s+2}$$

and

$$\frac{Y(s)}{U_2(s)} = \frac{1}{s^2+3s+2}$$

EXAMPLE 2-11 Consider a system with multiple inputs and multiple outputs. When the system has more than one output, the command

[NUM,den] = ss2tf(A,B,C,D,iu)

produces transfer functions for all outputs to each input. (The numerator coefficients are returned to matrix NUM with as many rows as there are outputs.)

Now consider the system defined by

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -25 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

This system involves two inputs and two outputs. Four transfer functions are involved: $Y_1(s)/U_1(s)$, $Y_2(s)/U_1(s)$, $Y_1(s)/U_2(s)$, and $Y_2(s)/U_2(s)$. (When considering input u_1, we assume that input u_2 is zero, and vice versa.) See the output of MATLAB Program 2–8.

MATLAB Program 2–8

```
>> A = [0   1;-25   -4];
>> B = [1   1;0   1];
>> C = [1   0;0   1];
>> D = [0   0;0   0];
>> [NUM,den] = ss2tf(A,B,C,D,1)

NUM =

     0    1     4
     0    0   -25

den =

     1    4    25

>> [NUM,den] = ss2tf(A,B,C,D,2)

NUM =

     0    1.0000     5.0000
     0    1.0000   -25.0000

den =

     1    4    25
```

This is the MATLAB representation of the following four transfer functions:

$$\frac{Y_1(s)}{U_1(s)} = \frac{s+4}{s^2+4s+25}, \quad \frac{Y_1(s)}{U_2(s)} = \frac{s+5}{s^2+4s+25}$$

$$\frac{Y_2(s)}{U_1(s)} = \frac{-25}{s^2+4s+25}, \quad \frac{Y_2(s)}{U_2(s)} = \frac{s-25}{s^2+4s+25}$$

Conversion from continuous time to discrete time. The command

[G,H] = c2d(A,B,Ts)

where Ts is the sampling period in seconds, converts the state-space model from continuous time to discrete time, assuming a zero-order hold on the inputs. That is, with that command,

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

is converted to

$$\mathbf{x}(k+1) = \mathbf{G}\mathbf{x}(k) + \mathbf{H}\mathbf{u}(k)$$

Consider, for example, the following continuous-time system:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -25 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

An equivalent discrete-time system can be obtained with the command [G,H] = c2d(A,B,Ts). If the sampling period Ts is assumed to be 0.05 sec, we have the result shown in MATLAB Program 2–9.

MATLAB Program 2–9

```
>> A = [0    1;-25    -4];
>> B = [0;1];
>> [G, H] = c2d(A, B, 0.05)

G =

      0.9709      0.0448
     -1.1212      0.7915

H =

      0.0012
      0.0448
```

Notice that if we used format long, we would have a more accurate **H** matrix. (See MATLAB Program 2–10.)

MATLAB Program 2–10

```
>> A = [0   1;-25   -4];
>> B = [0;1];
>> format long
>> [G, H] = c2d(A, B, 0.05)

G =

   0.97088325381929   0.04484704238264
  -1.12117605956599   0.79149508428874

H =

   0.00116466984723
   0.04484704238264
```

The discrete-time state-space equation when Ts = 0.05 sec is then given by

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 0.9709 & 0.04485 \\ -1.1212 & 0.7915 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0.001165 \\ 0.04485 \end{bmatrix} u(k)$$

If we used format short or if we did not specify format, we would have gotten

$$\mathbf{H} = \begin{bmatrix} 0.0012 \\ 0.0448 \end{bmatrix}$$

which is not quite accurate. (See MATLAB Program 2–9.) In transforming a continuous-time system to a discrete-time system, it is recommended that we use format long.

2–3 MATLAB REPRESENTATION OF SYSTEMS IN BLOCK DIAGRAM FORM

In this section, we shall review MATLAB representations of systems given in the form of block diagrams.

Figure 2–2(a) shows a block with a transfer function. Such a block represents a system or an element of a system. To simplify our presentation, we shall call the

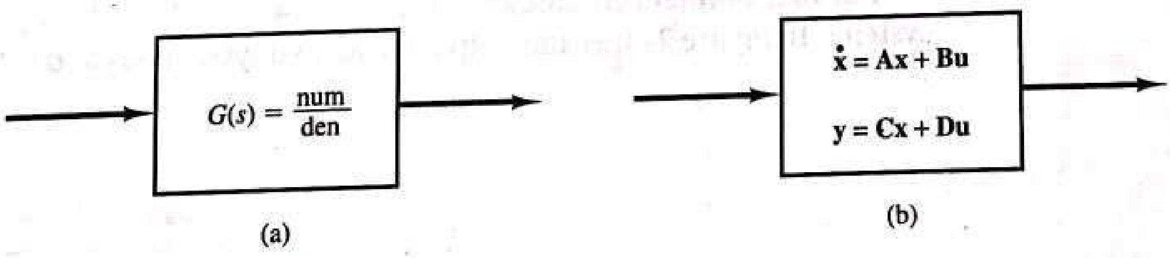


Figure 2–2
Block diagrams of systems.
(a) System is given by transfer function; (b) system is defined by state-space form.

block with a transfer function a system. MATLAB uses sys to represent such a system. The statement

sys = tf(num,den)

represents the system. Figure 2–2(b) shows a block with a state-space representation. The MATLAB representation of such a system is given by

sys = ss(A,B,C,D)

A physical system may involve many interconnected blocks. In what follows, we shall consider series-connected blocks, parallel-connected blocks, and feedback-connected blocks. Any linear time-invariant system may be represented by combinations of series-connected blocks, parallel-connected blocks, and feedback-connected blocks.

Series-connected blocks. In the system shown in Figure 2–3, G_1 and G_2 are series connected. System G_1 (which is represented by sys1) and system G_2 (which is represented by sys2) are respectively defined by

sys1 = tf(num1, den1)
sys2 = tf(num2, den2)

provided that the two systems are defined in terms of transfer functions. Then the series-connected system (sys) can be given by

sys = series(sys1, sys2)

The system's numerator and denominator can also be given by

[num, den] = series(num1,den1,num2,den2)

If systems G_1 and G_2 are given in state-space form, then their MATLAB representations are, respectively,

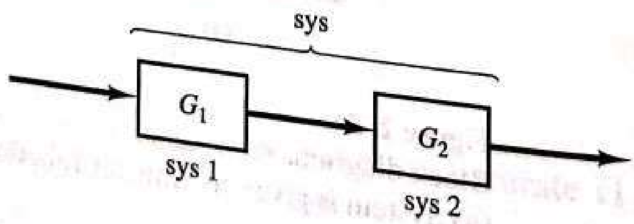
sys1 = ss(A1, B1, C1, D1)
sys2 = ss(A2, B2, C2, D2)

The series-connected system G_1G_2 is given by

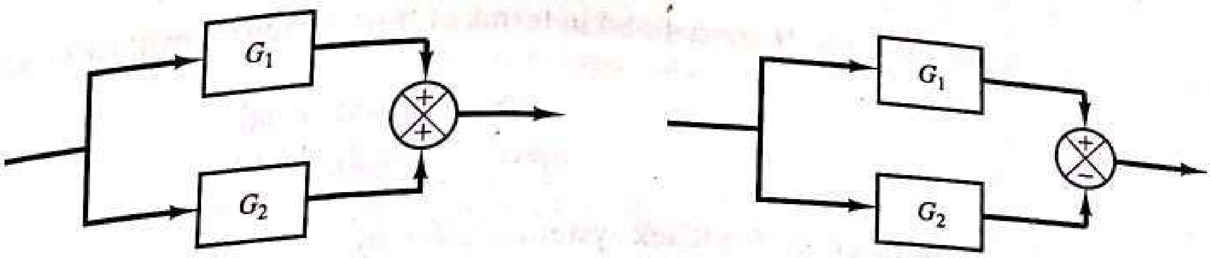
sys = series(sys1, sys2)

Parallel-connected blocks. Figures 2–4(a) and (b) show parallel-connected systems. In Figure 2–4(a), the outputs from two systems G_1 and G_2 are added, while

Figure 2–3
Series-connected blocks.



Scanned with CamScanner



(a) (b)

Figure 2–4
Parallel-connected systems.
(a) $G_1 + G_2$; (b) $G_1 - G_2$.

in Figure 2–4(b) the output from system G_2 is subtracted from the output of system G_1. If G_1 and G_2 are defined in terms of transfer functions, then

```
sys1 = tf(num1, den1)
sys2 = tf(num2, den2)
```

and the parallel-connected system $G_1 + G_2$ is given by

```
sys = parallel(sys1, sys2)
```

If G_1 and G_2 are in state-space form, then

```
sys1 = ss(A1, B1, C1, D1)
sys2 = ss(A2, B2, C2, D2)
```

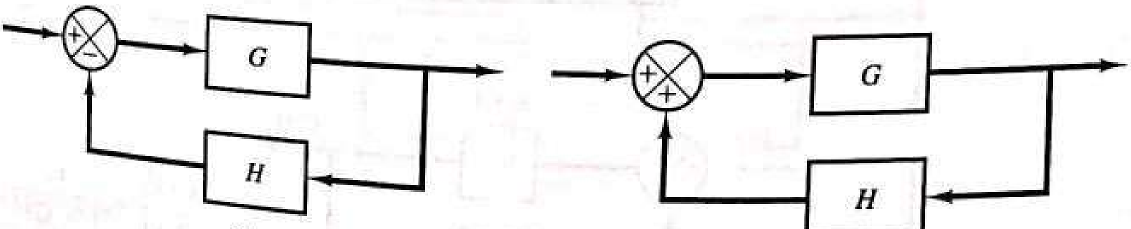
and the parallel-connected system $G_1 + G_2$ is given by

```
sys = parallel(sys1, sys2)
```

If the parallel-connected system is $G_1 - G_2$, as shown in Figure 2–4(b), then we define sys1 and sys2 as before, but change sys2 to −sys2 in the expressions for sys; that is,

```
sys = parallel(sys1, −sys2)
```

Feedback-connected blocks. Figure 2–5(a) shows a negative-feedback system and Figure 2–5(b) shows a positive-feedback system.



(a) (b)

Figure 2–5
(a) Negative-feedback system; (b) positive-feedback system.

If G and H are defined in terms of transfer functions, then

```
sysg = [numg, deng]
sysh = [numh, denh]
```

and the entire feedback system is given by

```
sys = feedback(sysg, sysh)
```

or

```
[num, den] = feedback(numg, deng, numh, denh)
```

If the system has a unity feedback function, then $H = [1]$ and sys can be given

```
sys = feedback(sysg, [1])
```

Note that, in treating the feedback system, MATLAB assumed that the feedback was negative. If the system involves a positive feedback, we need to add + in the argument of feedback as follows:

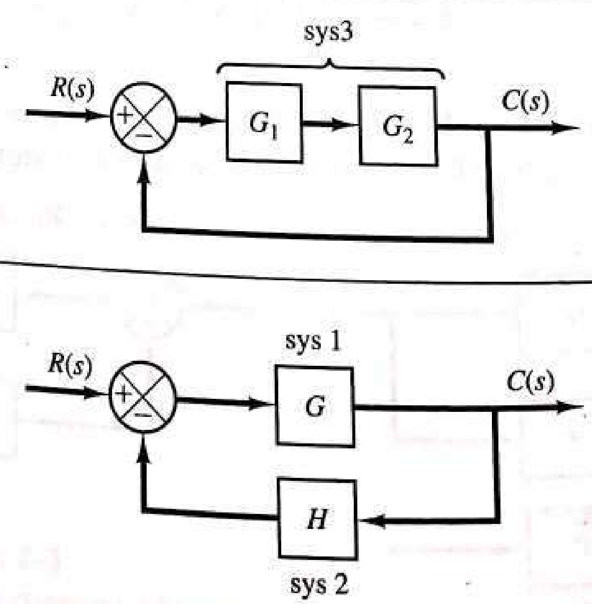
```
sys = feedback(sysg, sysh, +1)
```

Alternatively, we can use −sysh in the statement sys; that is,

```
sys = feedback(sysg, −sysh)
```

for the positive feedback system.

The following diagram summarizes the statement sys for feedback systems



$$\frac{C(s)}{R(s)} = \frac{G_1G_2}{1 + G_1G_2} = \frac{G_3}{1 + G_3}$$

where $G_3 = G_1G_2$

sys = feedback(sys3, [1])

$$\frac{C(s)}{R(s)} = \frac{G}{1 + GH}$$

sys = feedback(sys1, sys2)

EXAMPLE 2-12 Obtain the transfer functions of the cascaded system, parallel system, and feedback system shown in Figure 2–6. Assume that $G_1(s)$ and $G_2(s)$ are as follows:

$$G_1(s) = \frac{\text{num1}}{\text{den1}} = \frac{10}{s^2 + 2s + 10}, \qquad G_2(s) = \frac{\text{num2}}{\text{den2}} = \frac{5}{s + 5}$$

To obtain the transfer functions of the cascaded system, parallel system, or feedback (closed-loop) system, we use the following commands:

```
[num, den] = series(num1,den1,num2,den2)
[num, den] = parallel(num1,den1,num2,den2)
[num, den] = feedback(num1,den1,num2,den2)
```

MATLAB Program 2–11 gives $C(s)/R(s)$ = num/den for each arrangement of $G_1(s)$ and $G_2(s)$. Note that the command

```
printsys(num,den)
```

displays the num/den [that is, the transfer function $C(s)/R(s)$] of the system considered.

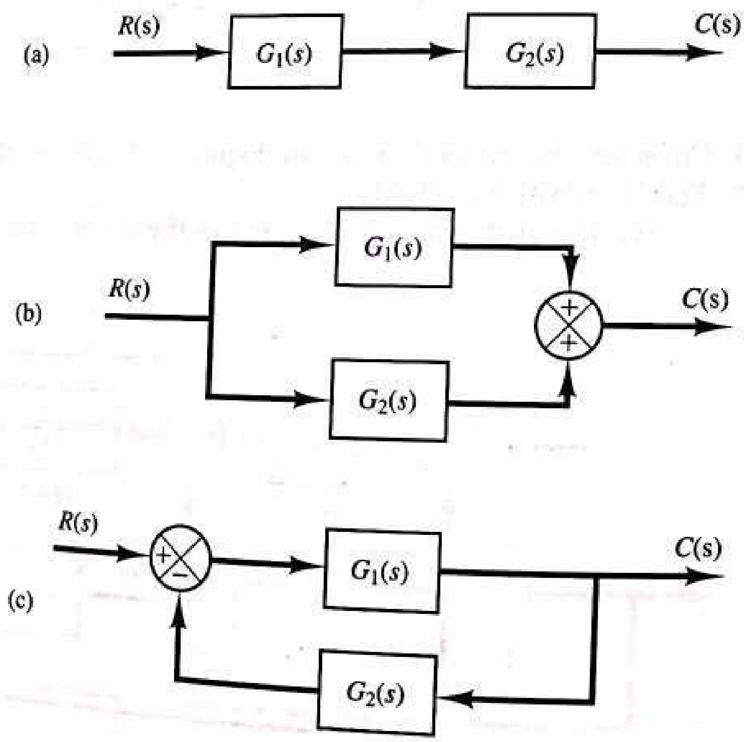


Figure 2–6
(a) Cascaded system;
(b) parallel system;
(c) feedback system.

MATLAB Program 2–11

```
>> num1 = [10];
>> den1 = [1   2   10];
>> num2 = [5];
>> den2 = [1   5];
>> [num, den] = series(num1,den1,num2,den2);
>> printsys(num,den)

num/den =
                 50
    s^3 + 7s^2 + 20s + 50

>> [num, den] = parallel(num1,den1,num2,den2);
>> printsys(num,den)

num/den =

      5s^2 + 20s + 100
    s^3 + 7s^2 + 20s + 50

>> [num, den] = feedback(num1,den1,num2,den2);
>> printsys(num,den)

num/den =

          10s + 50
    s^3 + 7s^2 + 20s + 100
```

EXAMPLE 2–13 Consider the system shown in Figure 2–7. Obtain the closed-loop transfer func $Y(s)/U(s)$ with MATLAB.

MATLAB Program 2–12 generates the closed-loop transfer function $Y(s)/U(s)$.

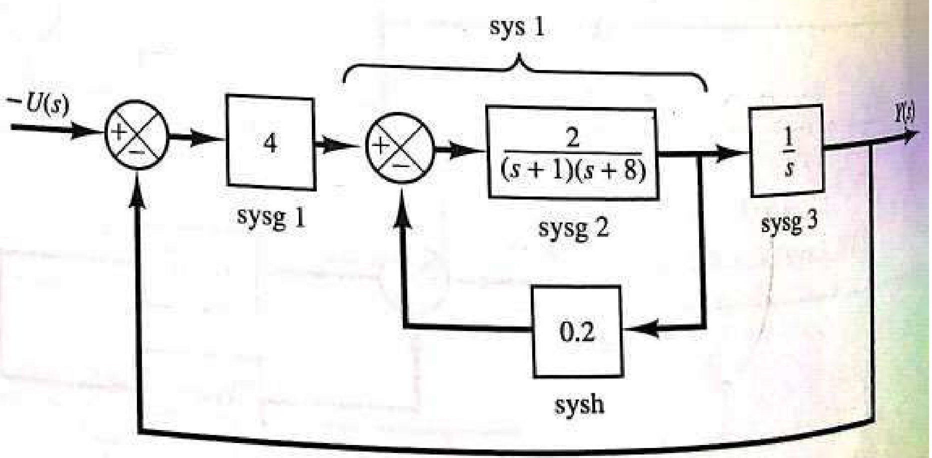


Figure 2–7
Closed-loop system.

MATLAB Program 2–12

```
>> sysg1 = [4];
>> numg2 = [2]; deng2 = [1   9   8]; sysg2 = tf(numg2,deng2);
>> numg3 = [1]; deng3 = [1   0]; sysg3 = tf(numg3,deng3);
>> sysh = [0.2];
>> sys1 = feedback (sysg2,sysh);
>> sys2 = series (sys1,sysg3);
>> sys3 = series (sysg1,sys2);
>> sys = feedback (sys3,[1])

Transfer function:
            8
--------------------------
s^3 + 9 s^2 + 8.4 s + 8
```

EXAMPLE 2–14 Consider the system shown in Figure 2–8. Obtain the closed-loop transfer function $Y(s)/U(s)$ with MATLAB. Also, obtain a state-space representation of the system.

MATLAB Program 2–13 produces the closed-loop transfer function $Y(s)/U(s)$ and also the state-space representation of the closed-loop system.

Note that in MATLAB Program 2–13 the state-space representation of the transfer-function system was obtained with the command

```
sys_ss = ss(sys)
```

where sys is given in terms of transfer functions.

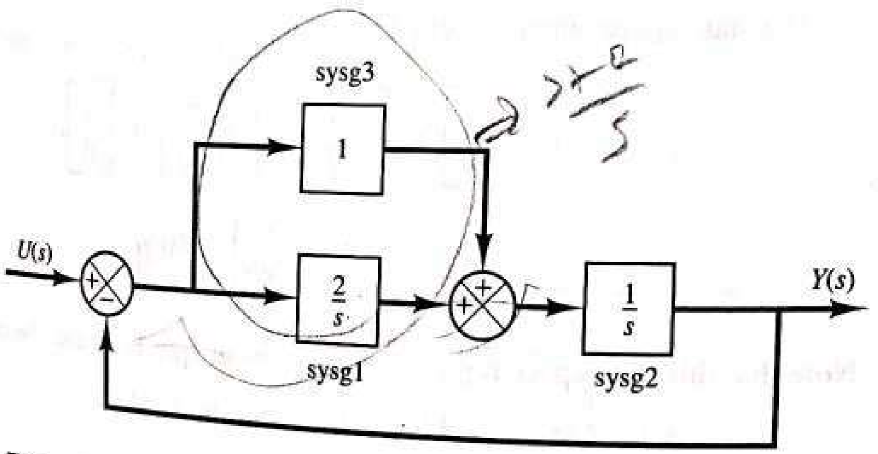


Figure 2–8
Closed-loop system.

MATLAB Program 2–13

```
>> numg1 = [2]; deng1 = [1   0]; sysg1 = tf(numg1,deng1);
>> numg2 = [1]; deng2 = [1   0]; sysg2 = tf(numg2,deng2);
>> sysg3 = [1];
>> sys1 = parallel(sysg1,sysg3);
>> sys2 = series(sys1,sysg2);
>> sys = feedback(sys2, [1])

Transfer function:
   s + 2
-----------
s^2 + s + 2

>> sys_ss = ss(sys)

a =
         x1   x2
    x1   -1   -1
    x2    2    0

b =
         u1
    x1    1
    x2    0

c =
         x1   x2
    y1    1    1

d =
         u1
    y1    0

Continuous-time model.
```

The state-space equations obtained are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -1 & -1 \\ 2 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}u \tag{2-9}$$

$$y = [1 \quad 1]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0]u \tag{2-10}$$

Note that this state-space representation is not unique. If we use the statement

[A,B,C,D] = tf2ss(num,den)

we obtain the following state-space equations:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -1 & -2 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}u \tag{2-11}$$

$$y = \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0]u \tag{2–12}$$

(See MATLAB Program 2–14.)

MATLAB Program 2–14

```
>> num = [1   2]; den = [1   1   2];
>> [A,B,C,D] = tf2ss(num,den)

A =

    -1   -2
     1    0

B =

     1
     0

C =

     1    2

D =

     0
```

The state-space equations [Equations (2–9) and (2–10)] can be transformed into those given by Equations (2–11) and (2–12) by means of the transformation matrix

$$\mathbf{P} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

If we use another transformation matrix

$$\mathbf{P} = \begin{bmatrix} 0 & 0.5 \\ 1 & -0.5 \end{bmatrix}$$

then the state-space equations [Equations (2–9) and (2–10)] become

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ -2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0]u$$

In fact, there are infinitely many state-space representations of the system shown in Figure 2–8.

EXAMPLE 2–15 Consider the system shown in Figure 2–9(a). Obtain a state-space representation of the closed-loop system.

One approach to obtaining a state-space representation of the system is to obtain the transfer-function expression for the closed-loop system and then transform that expression to a state-space expression. Another approach is to obtain state-space representations of individual blocks and then obtain the state-space representation of the entire closed-loop system.

In this example, we shall use the second approach. ~~Note that, to obtain a state-space expression of the transfer function in a block, it is necessary that the order of the numerator be less than or equal to the order of the denominator. Therefore, we redraw the system block diagram as shown in Figure 2–9(b).~~

MATLAB Program 2–15 produces the state-space representation of the closed-loop system shown in Figure 2–9(b). To obtain the corresponding transfer function, we use the statement

sys_tf = tf(sys_ss)

shown in MATLAB Program 2–15.

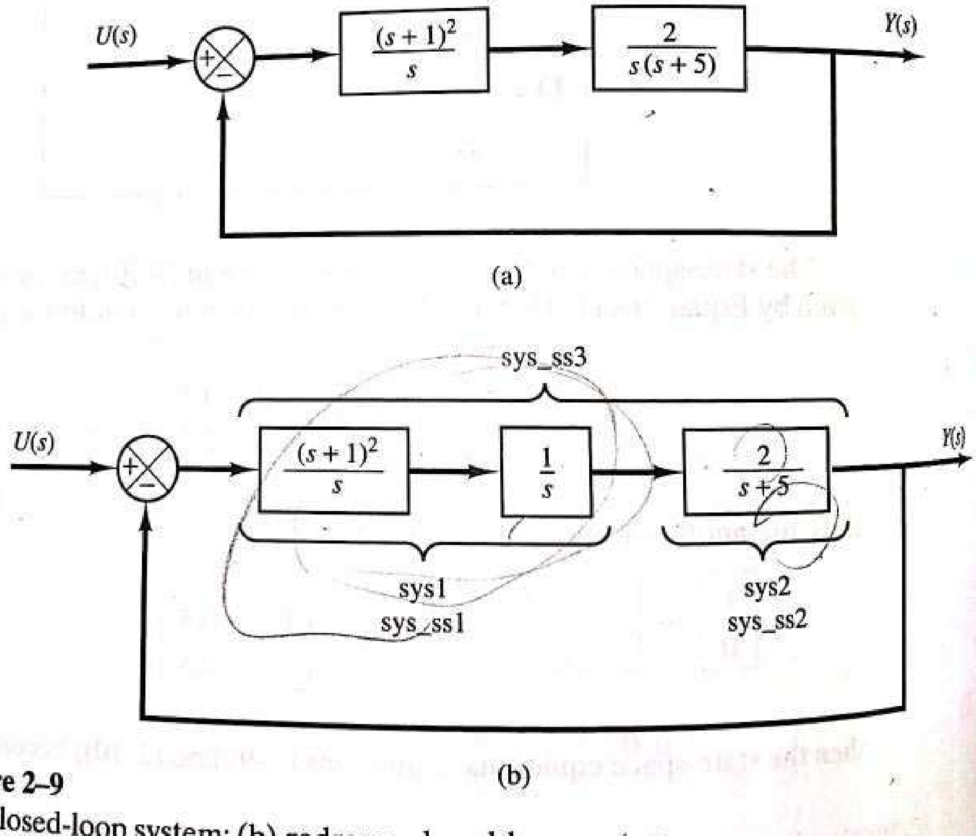


Figure 2–9
(a) Closed-loop system; (b) redrawn closed-loop system.

MATLAB Program 2–15

```
>> num1 = [1   2   1]; den1 = [1   0   0]; sys1 = tf(num1,den1);
>> num2 = [2]; den2 = [1   5]; sys2 = tf(num2,den2);
>> sys_ss1 = ss(sys1);
>> sys_ss2 = ss(sys2);
>> sys_ss3 = series(sys_ss1,sys_ss2);
>> sys_ss = feedback(sys_ss3, [1])

a =
          x1     x2     x3
   x1     -7      1      1
   x2     -4      0      0
   x3      0    0.5      0

b =
          u1
   x1      1
   x2      2
   x3      0

c =
          x1     x2     x3
   y1      2      0      0

d =
          u1
   y1      0

Continuous-time model.
>>
>> sys_tf = tf(sys_ss)

Transfer function:
   2 s^2 + 4 s + 2
---------------------
s^3 + 7 s^2 + 4 s + 2
```

Pole–zero cancellation. When we simplify a block diagram, the resulting closed-loop transfer function may involve pole–zero cancellation. If so, we would like to check the expression. If cancellation does occur, the cancelled expression can be obtained by use of the command

minreal

This command performs the pole–zero cancellation if there are common factors between the numerator and denominator and produces the minimal-order transfer function.

As an example, consider the following transfer function:

$$\frac{Y(s)}{U(s)} = \frac{(s+1)(s+2)}{(s+3)(s+4)(s+1)}$$

$$= \frac{s^2 + 3s + 2}{s^3 + 8s^2 + 19s + 12}$$

Clearly, $(s + 1)$ in the numerator and $(s + 1)$ in the denominator cancel each other. To obtain the minimal-order transfer function for $Y(s)/U(s)$, we may use the command minreal. (See MATLAB Program 2–16.)

MATLAB Program 2–16

```
>> num = [1  3  2]; den = [1  8  19  12]; sys = tf(num,den);
>> sys

Transfer function:
      s^2 + 3 s + 2
----------------------
s^3 + 8 s^2 + 19 s + 12

>> sys_min = minreal(sys)

Transfer function:
     s + 2
--------------
s^2 + 7 s + 12
```

The minimal-order transfer function (the canceled transfer function) is

$$\frac{Y(s)}{U(s)} = \frac{s+2}{s^2 + 7s + 12}$$