Appendix 3A
Basics of State Variable Modeling

The differential equations of a lumped linear network can be written in the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\,\boldsymbol{u}(t) \qquad (1)$$
$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\,\boldsymbol{u}(t)$$

This system of first-order differential equations is known as the *state equation* of the system and $\boldsymbol{x}(t)$ is the state vector and $\boldsymbol{u}(t)$ is the input vector. The second equation is referred to as the *output equation*. $\boldsymbol{A}$ is called the state matrix, $\boldsymbol{B}$ the input matrix, $\boldsymbol{C}$ the output matrix, and $\boldsymbol{D}$ the direct transition matrix. One advantage of the state-space method is that the form lends itself easily to the digital and/or analog computer methods of solution. Further, the state-space method can be easily extended to analysis of nonlinear systems. State equations may be obtained from an *n*th-order differential equation or directly from the system model by identifying appropriate state variables.

To illustrate how we select a set of state variables, consider an *n*th-order linear plant model described by the differential equation

$$\frac{d^n y}{dt^n} + a_{n-1}\frac{d^{n-1} y}{dt^{n-1}} + \cdots + a_1\frac{dy}{dt} + a_0 y = u(t) \qquad (2)$$

where $y(t)$ is the plant output and $u(t)$ is the plant input. A state model for this system is not unique but depends on the choice of a set of state variables. A useful set of state variables, referred to as *phase variables*, is defined as

$$x_1 = y, \quad x_2 = \dot{y}, \quad x_3 = \ddot{y}, \quad \cdots, \quad x_n = y^{n-1}$$

Taking the derivatives, we have

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = x_3, \quad \dot{x}_3 = x_4, \quad \cdots, \text{ and } \dot{x}_n \text{ is given by (2).} \qquad (3)$$
$$\dot{x}_n = -a_0 x_1 - a_1 x_2 - \cdots - a_{n-1} x_n + u(t)$$

or in matrix form

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 \\ -a_0 & -a_1 & -a_2 & & -a_{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u(t) \qquad (4)$$

and the output equation is

$$y = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \end{bmatrix} \boldsymbol{x} \qquad (5)$$

Example A.1

Obtain the state equation in phase variable form for the following differential equation.

$$2\frac{d^3y}{dt^3}+4\frac{d^2y}{dt^2}+6\frac{dy}{dt}+8y=10u(t)$$

The differential equation is third order, thus there are three state variables as follows

$$x_1=y,\quad x_2=\dot{y},\quad x_3=\ddot{y}$$

and the derivatives are

$$\dot{x}_1=x_2,\quad \dot{x}_2=x_3,\ \text{and}\ \ \dot{x}_3=-4x_1-3x_2-2x_3+5u(t)$$

Or in matrix form

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix}=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -4 & -3 & -2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}+\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}5u(t)$$

$$y=\begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}$$

The M-file **ode2phv.m** is developed which converts an *n*th-order ordinary differential equation to the state-space phase variable form. [***A, B, C***] = **ode2phv(*ai, k*)** returns the matrices ***A***, ***B***, ***C***, where ***ai*** is a row vector containing coefficients of the equation in descending order and ***k*** is the coefficient of the right-hand side.

```
ai = [2  4  6  8];
k = 10;
[A, B, C] = ode2phv(ai, k)
```

produces the following phase variable state representation

```
A =                B =          C =
    0    1    0        0              1    0    0
    0    0    1        0
   -4   -3   -2        5
```

Equations of Electrical Networks

The state variables are directly related to the energy-storage elements of a system. It would seem, therefore, that the number of independent initial conditions is equal to the number of energy-storing elements. This is true provided that there is no loop containing only capacitors and voltage sources and there is no cut set containing only inductive and current sources. In general, if there are n_c loops of all capacitors and voltage sources, and n_L cut sets of all inductors and current sources, the number of state variables is

$$n=e_L+e_C-n_C-n_L \tag{6}$$

where

e_L = number of inductors

e_C = number of capacitors

n_C = number of all capacitive and voltage source loops

n_L = number of all inductive and current source cut sets

Example A.2

Write the state equation for the network shown in Figure A.1.

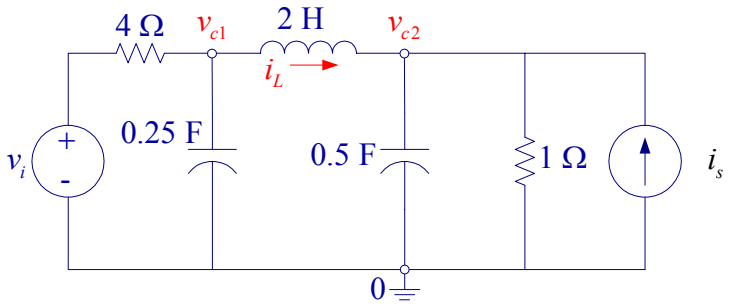


Figure A.1 Circuit of Example 2

Define the state variables as current through the inductor and voltage across the capacitors. Write two node equations containing capacitors and a loop equation containing the inductor. The state variables are v_{c1}, v_{c_2}, and i_L.

Node equations are

$$0.25\frac{dv_{c1}}{dt} + i_L + \frac{v_{c1} - v_i}{4} = 0 \quad \Rightarrow \quad \dot{v}_c = -v_{c1} - 4i_L + v_i$$

$$0.5\frac{dv_{c2}}{dt} - i_L + \frac{v_{c2}}{1} - i_s = 0 \quad \Rightarrow \quad \dot{v}_{c2} = -2v_{c1} + 2v_{c2} + 2i_s$$

and the loop equation is

$$2\frac{di_L}{dt} + v_{c2} - v_{c1} = 0 \quad \Rightarrow \quad \dot{i}_L = 0.5v_{c1} - 0.5v_{c2}$$

or

$$\begin{bmatrix} \dot{v}_{c1} \\ \dot{v}_{c2} \\ \dot{i}_L \end{bmatrix} = \begin{bmatrix} -1 & 0 & -4 \\ 0 & -2 & 2 \\ 0.5 & -0.5 & 0 \end{bmatrix} \begin{bmatrix} v_{c1} \\ v_{c2} \\ i_L \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 2 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_i \\ i_s \end{bmatrix}$$

Simulation Diagram

Equation (3) indicates that state variables are determined by integrating the corresponding state equation. A diagram known as the *simulation diagram* can be constructed to model the given differential equations. The basic element of the simulation diagram is the integrator. The first equation in (3) is

$$\dot{x}_1 = x_2$$

Integrating, we have

$$x_1 = \int x_2 dx$$

The above integral is represented by the time-domain diagram shown in Figure 2 (a) similar to the block diagram or the time-domain diagram shown in Figure 2 (b) similar to the signal flow graph.

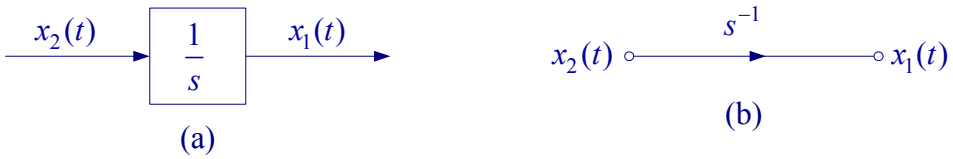


Figure A.2 Simulation diagram for integrator

It is important to know that although the symbol $1/s$ is used for integration, the simulation diagram is a time domain representation. The number of integrators is equal to the number of state variables. For example, for the state equation in Example 1 we have three integrators in cascade, the three state variables are assigned to the output of each integrator as shown in Figure 3. The last equation in (3) is represented via a summing point and feedback paths. Completing the output equation, the simulation diagram known as *phase-variable control canonical* form is obtained.

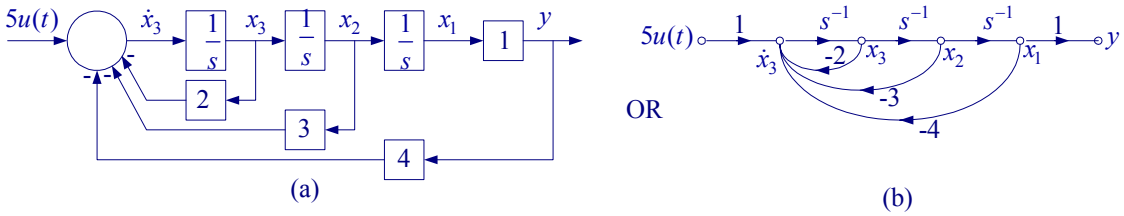


Figure A.3 Simulation diagram for Example 1

Transfer Function to State-Space Conversion
Direct Decomposition

Consider the transfer function of a third-order system

$$\frac{Y(s)}{U(s)} = \frac{b_2 s^2 + b_1 s + b_0}{s^3 + a_2 s^2 + a_1 s + a_0} \tag{7}$$

where the numerator degree is lower than that of the denominator. The above transfer function is decomposed into two blocks as shown in Figure 4.

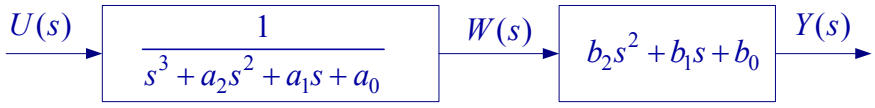


Figure A.4 Transfer function (7) arranged in cascade form

Denoting the output of the first block as $W(s)$, we have

$$W(s) = \frac{U(s)}{s^3 + a_2 s^2 + a_1 s + a_0} \qquad \text{and} \qquad Y(s) = b_2 s^2 W(s) + b_1 s W(s) + b_0 W(s)$$

or

$$s^3W(s) = -a_2 s^2 W(s) - a_1 s W(s) - a_0 W(s) + U(s)$$

This results in the following time-domain equation

$$\dddot{w} = -a_2\ddot{w} - a_1\dot{w} - a_0 w + u(t) \quad \text{and} \quad y(t) = b_2\ddot{w} + b_1\dot{w} + b_0 w$$

From the above expression we see that $\dddot{w}$ has to go through three integrators to get w as shown in Figure 5. Completing the above equations results in the phase-variable control canonical simulation diagram.

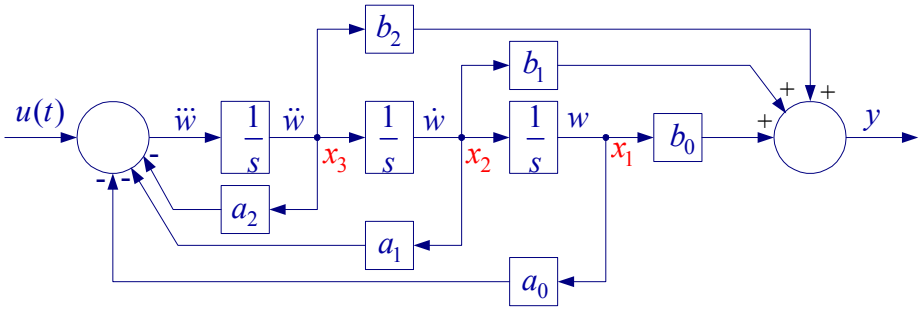


Figure A.5 Phase variable control canonical simulation diagram.

The above simulation in block diagram form is suitable for SIMULINK diagram construction. You may find it easier to construct the simulation diagram similar to the signal flow graph as shown in Figure 6.

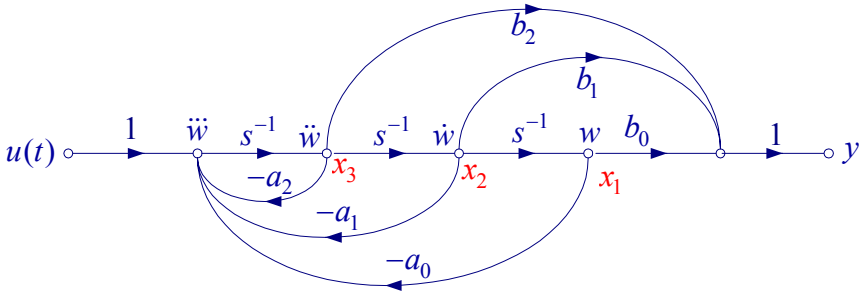


Figure A.6 Phase variable control canonical simulation diagram.

In order to write the state equation, the state variables $x_1(t)$, $x_2(t)$, and $x_3(t)$ are assigned to the output of each integrator from the right to the left. Next an equation is written for the input of each integrator. The results are

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = x_3$$
$$\dot{x}_3 = -a_0 x_1 - a_1 x_2 - a_0 x_1 + u(t)$$

and the output equation is

$$y = b_0 x_1 + b_1 x_2 + b_3 x_3$$

or in matrix form

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix}=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -a_0 & -a_1 & -a_2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}+\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}u(t) \qquad (8)$$

$$y=\begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}$$

It is important to note that the Mason's gain formula can be applied to the simulation diagram in Figure 6 to obtain the original transfer function. Indeed Δ of Mason's gain formula is the characteristic equation. Also, the determinant of $|sI-A|$ matrix in (8), results in the characteristics equation. Keep in mind that there is not a unique state space representation for a given transfer function.

The Control System Toolbox contains a set of functions for model conversion.
[***A, B, C, D***] = **tf2ss**(**num, den**) converts the system in transfer function from to state-space phase variable control canonical form.

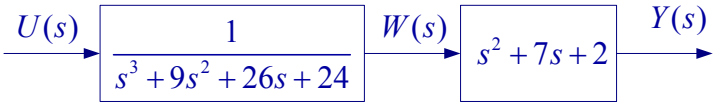
Example A.3

For the following transfer function

$$G(s)=\frac{Y(s)}{U(s)}=\frac{s^2+7s+2}{s^3+9s^2+26s+24}$$

(a) Draw the simulation diagram and find the state-space representation of the above transfer function.
(b) Use MATLAB Control System Toolbox [***A, B, C, D***] = **tf2ss**(**num, den**) to find the state model.

(a) Draw the transfer function block diagram in cascade form



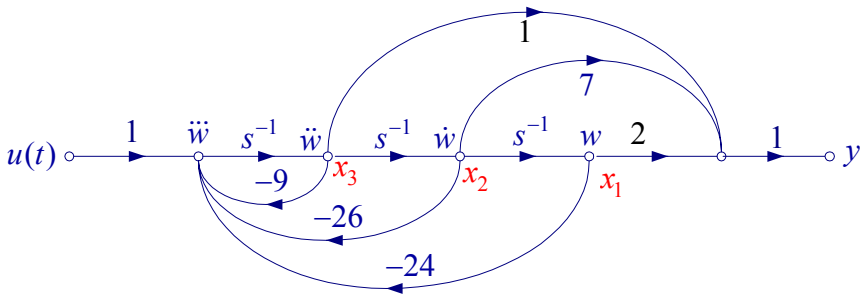
From this we have

$s^3W(s)=-9s^2W(s)-26sW(s)-24W(s)+U(s)$ & $Y(s)=s^2W(s)+7sW(s)+2W(s)$

or in time-domain

$\dddot{w}=-9\ddot{w}-26\dot{w}-24w+u$ & $y=\ddot{w}+7\dot{w}+2w$

The above time-domain equations yield the following simulation diagram



To obtain the state equation, the state variables $x_1(t)$, $x_2(t)$, and $x_3(t)$ are assigned to the output of each integrator from the right to the left. Next an equation is written for the input of each integrator. The results are

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = x_3$$
$$\dot{x}_3 = -24x_1 - 26x_2 - 9x_1 + u(t)$$

and the output equation is

$$y = 2x_1 + 7x_2 + x_3$$

or in matrix form

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -24 & -26 & -9 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u(t)$$

$$y = \begin{bmatrix} 2 & 7 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

(b) We write the following statements

```
num = [1  7  2]; den = [1   9   26   24];
[A, B, C, D] = tf2ss(num, den)
```

The result is

```
A =                          B =        C =                  D =
    -9   -26   -24               1          1   7   2            0
     1     0     0               0
     0     1     0               0
```

Note that MATLAB assigns x_1 to the output of the first integrator, and x_2, and x_3 to the output of the second and third integrators.

State-Space to Transfer Function Conversion

Consider the state and output equations

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\,u(t)$$

$$y = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\,u(t)$$

Taking the Laplace transform

$$sX(s) = AX(s) + BU(s) \qquad \Rightarrow \qquad [sI - A]X(s) = BU(s)$$

$$Y(s) = CX(s) + DU(s)$$

Substituting for $X(s)$ in the second equation above, we get

$$Y(s) = C[SI - A]^{-1}BU(s) + DU(s)$$

or

$$\frac{Y(s)}{U(s)} = C[SI - A]^{-1}B + D$$

In MATLAB [**num, den**] = **ss2tf**(***A, B, C, D, i***) converts the state equation to a transfer function for the *i*th input.

Example A.4

A system is described by the following state-space equations

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -6 & -5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

$$y = [8 \quad 1] \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix}$$

Obtain the system transfer function using the formula in (3.18)

$$[SI - A] = \begin{bmatrix} s & -1 \\ 6 & s+5 \end{bmatrix} \quad \Rightarrow \quad \Phi(s) = [SI - A]^{-1} = \frac{\begin{bmatrix} s+5 & 1 \\ 6 & s \end{bmatrix}}{s^2 + 5s + 6}$$

$$G(s) = C[SI - A]^{-1}B = [8 \quad 1] \frac{\begin{bmatrix} s+5 & 1 \\ 6 & s \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}}{s^2 + 5s + 6} = \frac{[8 \quad 1] \begin{bmatrix} 1 \\ s \end{bmatrix}}{s^2 + 5s + 6} = \frac{8 + s}{s^2 + 5s + 6}$$

Therefore

$$G(s) = \frac{s + 8}{s^2 + 5s + 6}$$

In MATLAB [**num, den**] = **ss2tf**(***A, B, C, D, i***) converts the state equation to a transfer function for the *i*th input.

Example A.5

A system is described by the following state-space equations

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix}=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -1 & -2 & -3\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}+\begin{bmatrix}10\\ 0\\ 0\end{bmatrix}u(t) \qquad (8)$$

$$y=\begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}$$

Find the transfer function, $G(s)=\dfrac{Y(s)}{U(s)}$. The following statements:

```
A = [0   1   0;  0   0   1; -1  -2  -3];  B = [10;  0;  0];
C = [1   0   0];  D = [0];
[num, den] = ss2tf(A, B, C, D, 1)
G = tf(num, den)
```

results in

```
num =
         0.0000   10.0000   30.0000   20.0000
den =
         1.0000    3.0000    2.0000    1.0000
Transfer function:
                    10 s^2 + 30 s + 20
                    ------------------
                    s^3 + 3 s^2 + 2 s + 1
```

Also, [z, p] = ss2tf(A, B, C, D, 1) converts the state equation to transfer function in factored form.

MALAB Control System Toolbox contains many functions for model creation and inversion, data extraction, and system interconnections. A few of these functions for continuous-time control systems are listed below. For a complete list of all functions type **help/control/control** at MATLAB prompt.

tf	Create transfer function models.
zpk	Create zero/pole/gain models.
ss	Create state-space models.
tfdata	Extract numerator(s) and denominator(s).
zpkdata	Extract zero/pole/gain data.
ssdata	Extract state-space matrices.
append	Group LTI systems by appending inputs and outputs.
parallel	Generalized parallel connection (see also overloaded +).
series	Generalized series connection (see also overloaded *).
feedback	Feedback connection of two systems.
connect	Derive state-space model from block diagram description.
blkbuild	Builds a model from a block diagram.

The Control System Toolbox supports four commonly used representations of linear time-invariant (LTI) systems: **tf**, **zpk**, and **ss** objects. To create an LTI model or object, use the corresponding constructor **tf**, **zpk**, or **ss**. For example,

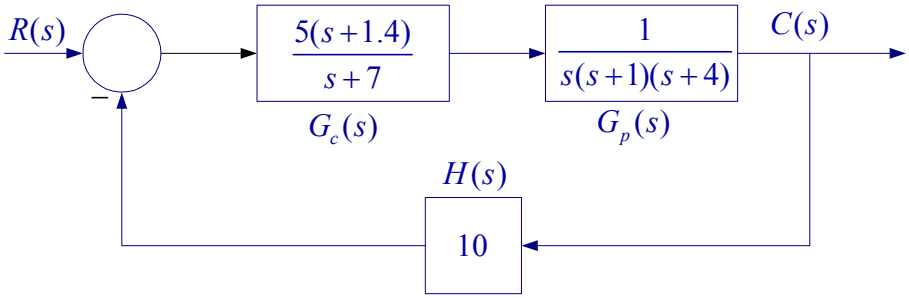
sys = tf(1,[1 0]) .

creates the transfer function H(s) = 1/s. The result sys is a **tf** object containing the numerator and denominator data. You can then manipulate the entire model as the single MATLAB variable sys. For more details and examples on how to specify the various types of LTI models, type **ltimodels** followed by **tf**, **zpk**, or **ss**.

The functions **tfdata**, **zpkdata**, and **ssdata** are provided for extracting the parameters of the **tf**, **zpk** and **ss** objects. For example the command **[num, den] = tfdata(T, 'v')** returns the numerator and denominator of the **tf** object. The argument **'v'** returns the numerator and denominator as row vectors rather than cell arrays. The Control System Toolbox contains seven more functions, which are useful for creating a single model out of its components.

Example A.6

Use **feedback function** to obtain the closed-loop transfer function and the **tf2ss** function to obtain the closed-loop state-space model



The following commands

```
Gc = tf(5*[1  1.4], [1  7])       % transfer function Gc
Gp = tf([1], [1 5 4 0]);          % transfer function Gp
H = 10;
G = series(Gc, Gp)                % connects Gc & Gp in cascade
T = feedback(G, H)                % obtains the closed loop transfer function
[num, den] = tfdata(T, 'v');      % returns num & den as row arrays
[A, B, C, D]=tf2ss(num, den)  % returns the A,B, C, D matrices
                                  % of the state space model
```

result in

Transfer function:

```
          5 s + 7
---------------------------------
s^4 + 12 s^3 + 39 s^2 + 78 s + 70
```

A =

-12	-39	-78	-70
1	0	0	0
0	1	0	0
0	0	1	0

B =

1
0
0
0

C =

[0	0	5	7]

D =

0

For analytical solution of state equation refer to the lecture notes on Chapter 3.