3.1 State Space Models

In this section we study state space models of continuous-time linear systems. The corresponding results for discrete-time systems, obtained via duality with the continuous-time models, are given in Section 3.3.

The state space model of a continuous-time dynamic system can be derived either from the system model given in the time domain by a differential equation or from its transfer function representation. Both cases will be considered in this section. Four state space forms—the phase variable form (controller form), the observer form, the modal form, and the Jordan form—which are often used in modern control theory and practice, are presented.

3.1.1 The State Space Model and Differential Equations

Consider a general nth-order model of a dynamic system represented by an nth-order differential equation

$$\begin{aligned}\frac{d^n y(t)}{dt^n} + a_{n-1}\frac{d^{n-1}y(t)}{dt^{n-1}} + \cdots + a_1\frac{dy(t)}{dt} + a_0 y(t) \\ = b_n\frac{d^n u(t)}{dt^n} + b_{n-1}\frac{d^{n-1}u(t)}{dt^{n-1}} + \cdots + b_1\frac{du(t)}{dt} + b_0 u(t)\end{aligned} \tag{3.1}$$

At this point we assume that all initial conditions for the above differential equation, i.e. $y(0^-), dy(0^-)/dt, ..., d^{n-1}y(0^-)/dt^{n-1}$, are equal to zero. We will show later how to take into account the effect of initial conditions.

In order to derive a systematic procedure that transforms a differential equation of order n to a state space form representing a system of n first-order differential equations, we first start with a simplified version of (3.5), namely we study the case when no derivatives with respect to the input are present

$$\frac{d^n y(t)}{dt^n} + a_{n-1}\frac{d^{n-1}y(t)}{dt^{n-1}} + \cdots + a_1\frac{dy(t)}{dt} + a_0 y(t) = u(t) \quad (3.2)$$

Introduce the following (easy to remember) change of variables

$$\begin{aligned} x_1(t) &= y(t) \\ x_2(t) &= \frac{dy(t)}{dt} \\ x_3(t) &= \frac{d^2y(t)}{dt^2} \\ &\vdots \\ x_n(t) &= \frac{d^{n-1}y(t)}{dt^{n-1}} \end{aligned} \quad (3.3)$$

which after taking derivatives leads to

$$\begin{aligned} \frac{dx_1(t)}{dt} &= \dot{x}_1 = \frac{dy(t)}{dt} = x_2(t) \\ \frac{dx_2(t)}{dt} &= \dot{x}_2 = \frac{d^2y(t)}{dt^2} = x_3(t) \\ \frac{dx_3(t)}{dt} &= \dot{x}_3 = \frac{d^3y(t)}{dt^3} = x_4(t) \\ &\vdots \\ \frac{dx_n(t)}{dt} &= \dot{x}_n = \frac{d^n y(t)}{dt^n} \end{aligned}$$

$$= -a_0 y(t) - a_1\frac{dy(t)}{dt} - a_2\frac{d^2y(t)}{dt^2} - \cdots - a_{n-1}\frac{d^{n-1}y(t)}{dt^{n-1}} + u(t)$$
$$= -a_0 x_1(t) - a_1 x_2(t) - \cdots - a_2 x_3(t) - \cdots - a_{n-1}x_n(t) + u(t)$$

$$(3.4)$$

The state space form of (3.8) is given by

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \cdots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & \cdots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & \cdots & -a_{n-1} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ \vdots \\ x_{n-1}(t) \\ x_n(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ 1 \end{bmatrix} u(t) \tag{3.5}$$

with the corresponding output equation obtained from (3.7) as

$$y(t) = [1 \quad 0 \quad \cdots \quad 0] \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_{n-1}(t) \\ x_n(t) \end{bmatrix} \tag{3.6}$$

The state space form (3.9) and (3.10) is known in the literature as the *phase variable canonical form.*

In order to extend this technique to the general case defined by (3.5), which includes derivatives with respect to the input, we form an auxiliary differential equation of (3.5) having the form of (3.6) as

$$\frac{d^n\xi(t)}{dt^n} + a_{n-1}\frac{d^{n-1}\xi(t)}{dt^{n-1}} + \cdots + a_1\frac{d\xi(t)}{dt} + a_0\xi(t) = u(t) \tag{3.7}$$

for which the change of variables (3.7) is applicable

$$\begin{aligned} x_1(t) &= \xi(t) \\ x_2(t) &= \frac{d\xi(t)}{dt} \\ x_3(t) &= \frac{d^2\xi(t)}{dt^2} \\ &\vdots \\ x_n(t) &= \frac{d^{n-1}\xi(t)}{dt^{n-1}} \end{aligned} \tag{3.8}$$

and then apply the superposition principle to (3.5) and (3.11). Since $\xi(t)$ is the response of (3.11), then by the superposition property the response of (3.5) is given by

$$y(t) = b_0\xi(t) + b_1\frac{d\xi(t)}{dt} + b_2\frac{d^2\xi(t)}{dt^2} + \cdots + b_n\frac{d^n\xi(t)}{dt^n} \tag{3.9}$$

Equations (3.12) produce the state space equations in the form already given by (3.9). The output equation can be obtained by eliminating $d^n\xi(t)/dt^n$ from (3.13), by using (3.11), that is

$$\frac{d^n\xi(t)}{dt^n} = u(t) - a_{n-1}x_n(t) - \cdots - a_1x_2(t) - a_0x_1(t)$$

This leads to the output equation

$$y(t) = \left[(b_0 - a_0b_n) \quad (b_1 - a_1b_n) \quad \cdots \quad (b_{n-1} - a_{n-1}b_n)\right]\begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} + b_nu(t) \tag{3.10}$$

It is interesting to point out that for $b_n = 0$, which is almost always the case, the output equation also has an easy-to-remember form given by

$$y(t) = [b_0 \quad b_1 \quad \cdots \quad b_{n-1}] \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} \tag{3.11}$$

Thus, in summary, for a given dynamic system modeled by differential equation (3.5), one is able to write immediately its state space form, given by (3.9) and (3.15), just by identifying coefficients a_i and b_i, $i = 0, 1, 2, ..., n-1$, and using them to form the corresponding entries in matrices $\mathbf{A}$ and $\mathbf{C}$.

Example 3.1: Consider a dynamical system represented by the following differential equation

$$y^{(6)} + 6y^{(5)} - 2y^{(4)} + y^{(2)} - 5y^{(1)} + 3y = 7u^{(3)} + u^{(1)} + 4u$$

where $y^{(i)}$ stands for the ith derivative, i.e. $y^{(i)} = d^i y/dt^i$. According to (3.9) and (3.14), the state space model of the above system is described by the following matrices

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -3 & 5 & -1 & 0 & 2 & -6 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\mathbf{C} = [4 \quad 1 \quad 0 \quad 7 \quad 0 \quad 0], \quad \mathbf{D} = 0$$

$\diamond$

3.1.2 State Space Variables from Transfer Functions

In this section, we present two methods, known as direct and parallel programming techniques, which can be used for obtaining state space models from system transfer functions. For simplicity, like in the previous subsection, we consider only single-input single-output systems.

The resulting state space models may or may not contain all the modes of the original transfer function, where by transfer function modes we mean poles of the original transfer function (before zero-pole cancellation, if any, takes place). If some zeros and poles in the transfer function are cancelled, then the resulting state space model will be of reduced order and the corresponding modes will not appear in the state space model. This problem of system reducibility will be addressed in detail in Chapter 5 after we have introduced the system controllability and observability concepts.

In the following, we first use direct programming techniques to derive the state space forms known as the controller canonical form and the observer canonical form; then, by the method of parallel programing, the state space forms known as modal canonical form and Jordan canonical form are obtained.

The Direct Programming Technique and Controller Canonical Form

This technique is convenient in the case when the plant transfer function is given in a nonfactorized polynomial form

$$\frac{Y(s)}{U(s)} = \frac{b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} \tag{3.12}$$

For this system an auxiliary variable $V(s)$ is introduced such that the transfer function is split as

$$\frac{V(s)}{U(s)} = \frac{1}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} \tag{3.13a}$$

$$\frac{Y(s)}{V(s)} = b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0 \tag{3.13b}$$

The block diagram for this decomposition is given in Figure 3.1.

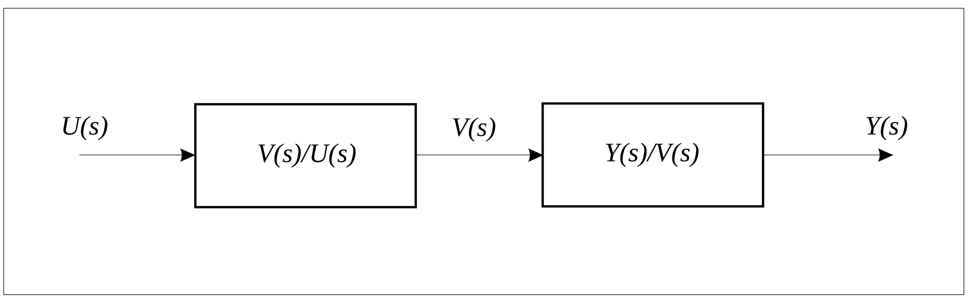

Figure 3.1: Block diagram representation for (3.17)

Equation (3.17a) has the same structure as (3.6), after the Laplace transformation is applied, which directly produces the state space system equation identical to (3.9). It remains to find matrices for the output equation (3.2). Equation (3.17b) can be rewritten as

$$Y(s) = b_n s^n V(s) + b_{n-1}s^{n-1}V(s) + \cdots + b_1 sV(s) + b_0 V(s) \tag{3.14}$$

indicating that $y(t)$ is just a superposition of $v(t)$ and its derivatives. Note that (3.17) may be considered as a differential equation in the operator form for zero initial conditions, where $s \equiv d/dt$. In that case, $V(s)$, $Y(s)$, and $U(s)$ are simply replaced with $v(t)$, $y(t)$, and $u(t)$, respectively.

The common procedure for obtaining state space models from transfer functions is performed with help of the so-called transfer function *simulation diagrams*. In the case of continuous-time systems, the simulation diagrams are elementary analog computers that solve differential equations describing systems dynamics. They are composed of integrators, adders, subtracters, and multipliers, which are physically realized by using operational amplifiers. In addition, function generators are used to generate input signals. The number of integrators in a simulation diagram is equal to the order of the differential equation under consideration. It is relatively easy to draw (design) a simulation diagram. There are many ways to draw a simulation diagram for a given dynamic system, and there are also many ways to obtain the state space form from the given simulation diagram.

The simulation diagram for the system (3.17) can be obtained by direct programming technique as follows. Take n integrators in cascade and denote their inputs, respectively, by $v^{(n)}(t), v^{(n-1)}(t), ..., v^{(1)}(t), v(t)$. Use formula (3.18) to construct $y(t)$, i.e. multiply the corresponding inputs $v^{(i)}(t)$ to integrators by the corresponding coefficients b_i and add them using an adder (see the top half of Figure 3.2, where $1/s$ represents the integrator block). From (3.17a) we have that

$$v^{(n)}(t) = u(t) - a_{n-1}v^{(n-1)}(t) - \cdots - a_1 v^{(1)}(t) - a_0 v(t)$$

which can be physically realized by using the corresponding feedback loops in the simulation diagram and adding them as shown in the bottom half of Figure 3.2.

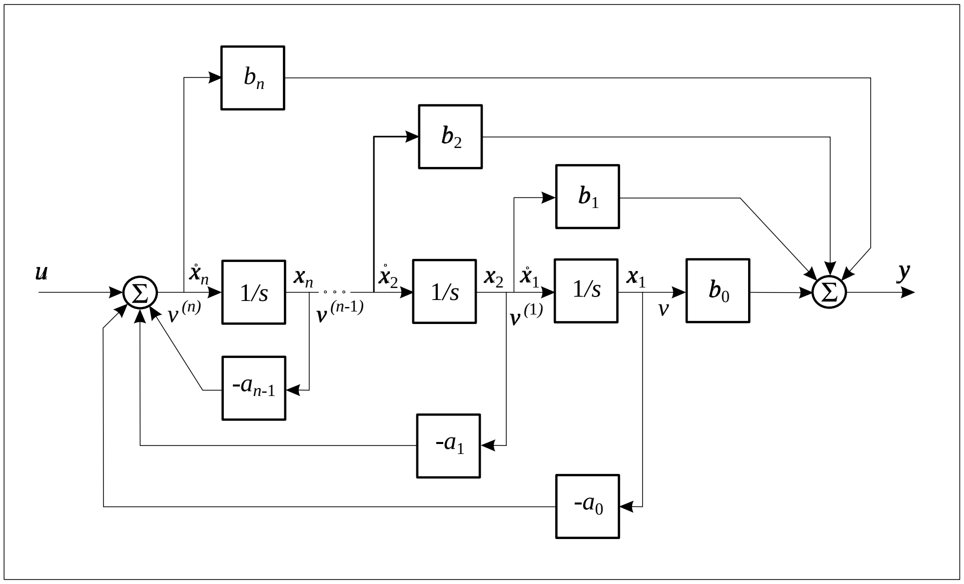

Figure 3.2: Simulation diagram for the direct programming technique (controller canonical form)

A systematic procedure to obtain the state space form from a simulation diagram is *to choose the outputs of integrators as state variables*. Using this convention, the state space model for the simulation diagram presented in Figure 3.2 is obtained in a straightforward way by reading and recording information from the simulation diagram, which leads to

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & \cdots & -a_{n-1} \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ 1 \end{bmatrix} u(t) \tag{3.15}$$

and

$$y(t) = \left[(b_0 - a_0 b_n) \quad (b_1 - a_1 b_n) \quad \cdots \quad (b_{n-1} - a_{n-1} b_n)\right]\mathbf{x}(t) \\ + b_n u(t) \tag{3.16}$$

This form of the system model is called the *controller canonical form*. It is identical to the one obtained in the previous section—equations (3.9) and (3.14). Controller canonical form plays an important role in control theory since it represents the so-called controllable system. System controllability is one of the main concepts of modern control theory. It will be studied in detail in Chapter 5.

It is important to point out that there are many state space forms for a given dynamical system, and that all of them are related by linear transformations. More about this fact, together with the development of other important state space canonical forms, can be found in Kailath (1980; see also similarity transformation in Section 3.4).

Note that the MATLAB function `tf2ss` produces the state space form for a given transfer function, in fact, it produces the controller canonical form.

Example 3.2: The transfer function of the flexible beam from Section 2.6 is given by

$$G(s) = \frac{1.65s^4 - 0.331s^3 - 576s^2 + 90.6s + 19080}{s^6 + 0.996s^5 + 463s^4 + 97.8s^3 + 12131s^2 + 8.11s}$$

Using the direct programming technique with formulas (3.19) and

(3.20), the state space controller canonical form is given by

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -8.11 & -12131 & -97.8 & -463 & -0.996 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u$$

and

$$y = [19080 \quad 90.6 \quad -576 \quad -0.331 \quad 1.65 \quad 0]\mathbf{x}$$

◇

Direct Programming Technique and Observer Canonical Form

In addition to controller canonical form, *observer canonical form* is related to another important concept of modern control theory: system observability. Observer canonical form has a very simple structure and represents an observable system. The concept of linear system observability will be considered thoroughly in Chapter 5.

Observer canonical form can be derived as follows. Equation (3.16) is written in the form

$$\begin{aligned} Y(s)\big(s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0\big) \\ = U(s)\big(b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0\big) \end{aligned} \tag{3.17}$$

and expressed as

$$\begin{aligned} Y(s) = -\frac{1}{s^n}\big(a_{n-1}s^{n-1} + \cdots + a_1 s + a_0\big)Y(s) \\ +\frac{1}{s^n}U(s)\big(b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0\big) \end{aligned} \tag{3.18}$$

leading to

$$Y(s) = -\frac{1}{s}a_{n-1}Y(s) - \frac{1}{s^2}a_{n-2}Y(s) - \cdots - \frac{1}{s^{n-1}}a_1Y(s)$$
$$-\frac{1}{s^n}a_0Y(s) + b_nU(s) + \frac{1}{s}b_{n-1}U(s) + \frac{1}{s^2}b_{n-2}U(s) +$$
$$\cdots + \frac{1}{s^{n-1}}b_1U(s) + \frac{1}{s^n}b_0U(s) \tag{3.19}$$

This relationship can be implemented by using a simulation diagram composed of n integrators in a cascade, and letting the corresponding signals to pass through the specified number of integrators. For example, terms containing $1/s$ should pass through only one integrator, signals $a_{n-2}y(t)$ and $b_{n-2}u(t)$ should pass through two integrators, and so on. Finally, signals $a_0y(t)$ and $b_0u(t)$ should be integrated n-times, i.e. they must pass through all n integrators. The corresponding simulation diagram is given in Figure 3.3.

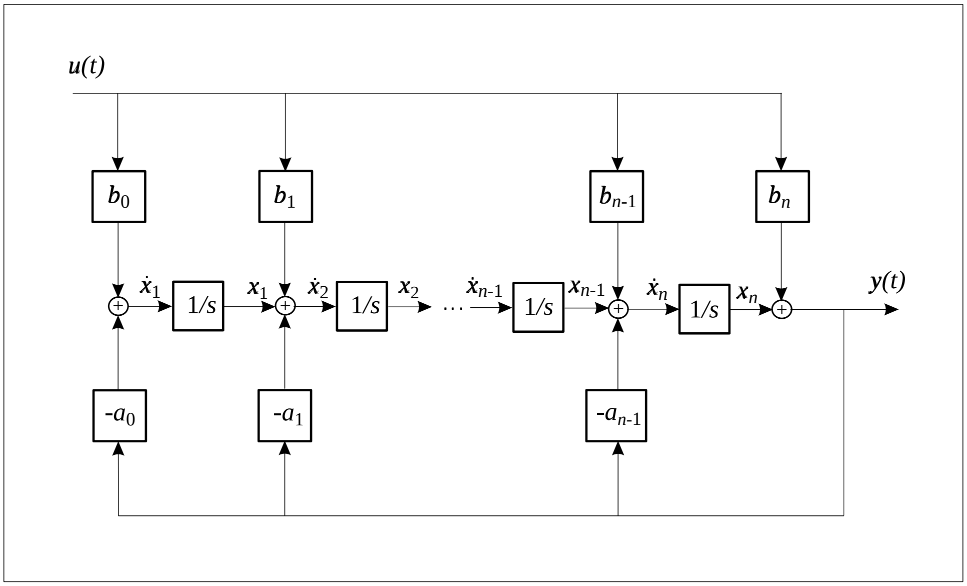

Figure 3.3: Simulation diagram for observer canonical form

Defining the state variables as the outputs of integrators, and recording relationships among state variables and the system output, we get from the above figure

$$y(t) = x_n(t) + b_n u(t) \tag{3.20}$$

$$\begin{aligned}
\dot{x}_1(t) &= -a_0 y(t) + b_0 u(t) = -a_0 x_n(t) + (b_0 - a_0 b_n)u(t) \\
\dot{x}_2(t) &= -a_1 y(t) + b_1 u(t) + x_1(t) \\
&= x_1(t) - a_1 x_n(t) + (b_1 - a_1 b_n)u(t) \\
\dot{x}_3(t) &= -a_2 y(t) + b_2 u(t) + x_2(t) \\
&= x_2(t) - a_2 x_n(t) + (b_2 - a_2 b_n)u(t) \\
&\cdots \\
\dot{x}_n(t) &= -a_{n-1} y(t) + b_{n-1} u(t) + x_{n-1}(t) \\
&= x_{n-1}(t) - a_{n-1} x_n(t) + (b_{n-1} - a_{n-1} b_n)u(t)
\end{aligned} \tag{3.21}$$

The matrix form of observer canonical form is easily obtained from (3.24) and (3.25) as

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 0 & \dots & \dots & 0 & -a_0 \\ 1 & 0 & \dots & \dots & 0 & -a_1 \\ 0 & 1 & \ddots & \dots & \vdots & -a_2 \\ \vdots & 0 & 1 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 & -a_{n-2} \\ 0 & 0 & \dots & 0 & 1 & -a_{n-1} \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} b_0 - a_0 b_n \\ b_1 - a_1 b_n \\ b_2 - a_2 b_n \\ \vdots \\ \vdots \\ b_{n-1} - a_{n-1} b_n \end{bmatrix} u(t) \tag{3.22}$$

and

$$y(t) = [0 \quad \cdots \quad 0 \quad 1\,]\mathbf{x}(t) + b_n u(t) \tag{3.23}$$

Example 3.3: The observer canonical form for the flexible beam from Example 3.2 is given by

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -8.11 \\ 0 & 1 & 0 & 0 & 0 & -12131 \\ 0 & 0 & 1 & 0 & 0 & -97.8 \\ 0 & 0 & 0 & 1 & 0 & -463 \\ 0 & 0 & 0 & 0 & 1 & -0.996 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 19080 \\ 90.6 \\ -576 \\ -0.331 \\ 1.65 \\ 0 \end{bmatrix} u$$

and

$$y = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1\,]\mathbf{x}$$

$\diamond$

Observer canonical form is very useful for computer simulation of linear dynamical systems since it allows the effect of the system initial conditions to be taken into account. Thus, this form represents an observable system, in the sense to be defined in Chapter 5, which means that all state variables have an impact on the system output, and vice versa, that from the system output and state equations one is able to reconstruct the state variables at any time instant, and of course at zero, and thus, determine $x_1(0), x_2(0), ..., x_n(0)$ in terms of the original initial conditions $y(0^-), dy(0^-)/dt, ..., d^{n-1}y(0^-)/dt^{n-1}$. For more details see Section 5.5 for a subtopic on the observability role in analog computer simulation.

Parallel Programming Technique

For this technique we distinguish two cases: distinct real roots and multiple real roots of the system transfer function denominator.

Distinct Real Roots

This state space form is very convenient for applications. Derivation of this type of the model starts with the transfer function in the partial fraction expansion form. Let us assume, without loss of generality, that the polynomial in the numerator has degree of $m < n$, then

$$\begin{aligned} \frac{Y(s)}{U(s)} &= \frac{P_m(s)}{(s+p_1)(s+p_2)\cdots(s+p_n)} \\ &= \frac{k_1}{s+p_1} + \frac{k_2}{s+p_2} + \cdots + \frac{k_n}{s+p_n} \end{aligned} \tag{3.24}$$

Here $p_1, p_2, ..., p_n$ are *distinct real* roots (poles) of the transfer function denominator.

The simulation diagram of such a form is shown in Figure 3.4.

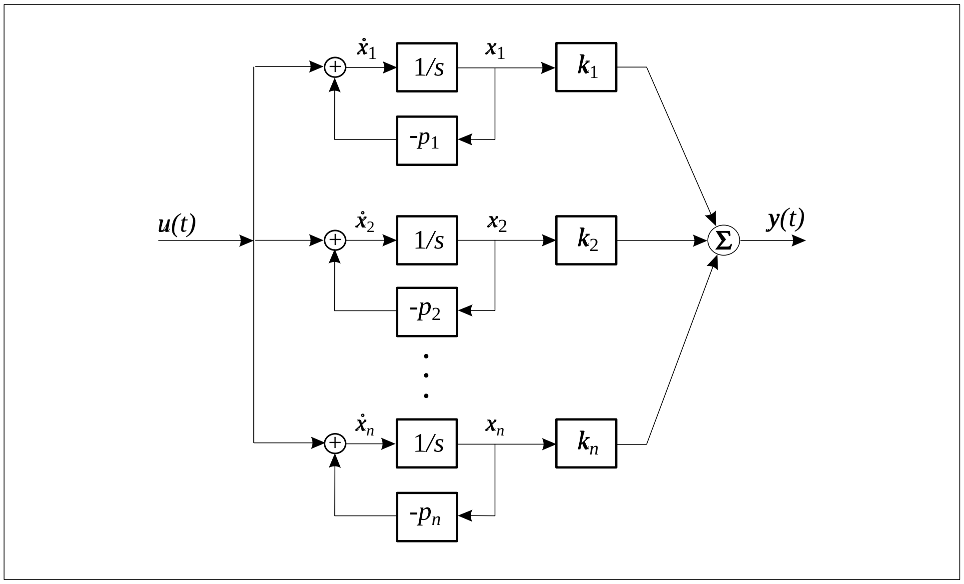

Figure 3.4: The simulation diagram for the parallel programming technique (modal canonical form)

The state space model derived from this simulation diagram is given by

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -p_1 & 0 & \cdots & \cdots & 0 \\ 0 & -p_2 & \ddots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & -p_n \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ 1 \end{bmatrix} u(t) \tag{3.25}$$

$$y(t) = [k_1 \quad k_2 \quad \cdots \quad \cdots \quad k_n]\mathbf{x}(t)$$

This form is known in the literature as the *modal canonical form* (also known as uncoupled form).

Example 3.4: Find the state space model of a system described by the transfer function

$$\frac{Y(s)}{U(s)} = \frac{(s+5)(s+4)}{(s+1)(s+2)(s+3)}$$

using both direct and parallel programming techniques.

The nonfactorized transfer function is

$$\frac{Y(s)}{U(s)} = \frac{s^2 + 9s + 20}{s^3 + 6s^2 + 11s + 6}$$

and the state space form obtained by using (3.19) and (3.20) of the direct programming technique is

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -11 & -6 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$

$$y = [20 \quad 9 \quad 1]\mathbf{x}$$

Note that the MATLAB function `tf2ss` produces

$$\underline{\dot{\mathbf{x}}} = \begin{bmatrix} -6 & -11 & -6 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \underline{\mathbf{x}} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u$$

$$y = [1 \quad 9 \quad 20]\underline{\mathbf{x}}$$

which only indicates a permutation in the state space variables, that is

$$\mathbf{x} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \underline{\mathbf{x}}$$

Employing the partial fraction expansion (which can be obtained by the MATLAB function `residue`), the transfer function is written as

$$\frac{Y(s)}{U(s)} = \frac{(s+5)(s+4)}{(s+1)(s+2)(s+3)} = \frac{6}{s+1} - \frac{6}{s+2} + \frac{1}{s+3}$$

The state space model, directly written using (3.29), is

$$\dot{\mathbf{x}} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -3 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 6 & -6 & 1 \end{bmatrix} \mathbf{x}$$

◇

Note that the parallel programming technique presented is valid only for the case of real distinct roots. If complex conjugate roots appear they should be combined in pairs corresponding to the second-order transfer functions, which can be independently implemented as demonstrated in the next example.

Example 3.5: Let a transfer function containing a pair of complex conjugate roots be given by

$$G(s) = \frac{4}{s+1-j} + \frac{4}{s+1+j} + \frac{2}{s+5} + \frac{3}{s+10}$$

We first group the complex conjugate poles in a second-order transfer function, that is

$$G(s) = \frac{8s+8}{s^2+2s+2} + \frac{2}{s+5} + \frac{3}{s+10}$$

Then, distinct real poles are implemented like in the case of parallel programming. A second-order transfer function, corresponding to the pair of complex conjugate poles, is implemented using direct programming, and added in parallel to the first-order transfer functions corresponding to the real poles. The simulation diagram is given in Figure 3.5, where the controller canonical form is used to represent a second-order transfer function corresponding to the complex conjugate poles. From this simulation diagram we have

$$\begin{aligned}
\dot{x}_1 &= -5x_1 + u \\
\dot{x}_2 &= -10x_2 + u \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -2x_3 - 2x_4 + u \\
y &= 2x_1 + 3x_2 + 8x_3 + 8x_4
\end{aligned}$$

so that the required state space form is

$$\dot{\mathbf{x}} = \begin{bmatrix} -5 & 0 & 0 & 0 \\ 0 & -10 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -2 & -2 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 2 & 3 & 8 & 8 \end{bmatrix} \mathbf{x}$$

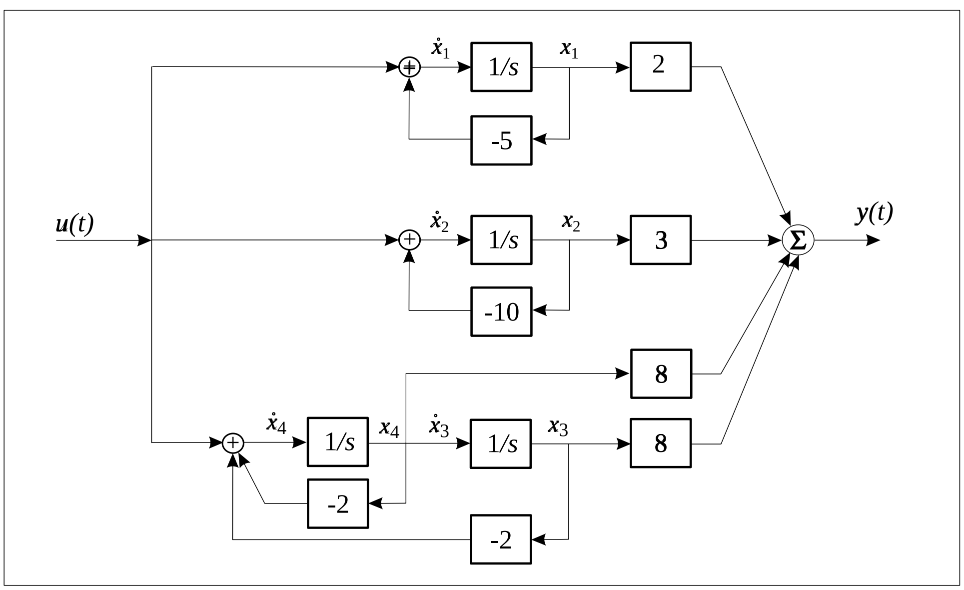

Figure 3.5: Simulation diagram for a system with complex conjugate poles

◇

Multiple Real Roots

When the transfer function has multiple real poles, it is not possible to represent the system in uncoupled form. Assume that a real pole p_1 of the transfer function has multiplicity r and that the other poles are real and distinct, that is

$$\frac{Y(s)}{U(s)} = \frac{N(s)}{(s+p_1)^r(s+p_{r+1})\cdots(s+p_n)}$$

The partial fraction form of the above expression is

$$\frac{Y(s)}{U(s)} = \frac{k_{11}}{s+p_1} + \frac{k_{12}}{(s+p_1)^2} + \cdots + \frac{k_{1r}}{(s+p_1)^r} + \frac{k_{r+1}}{s+p_{r+1}} + \cdots + \frac{k_n}{s+p_n}$$

The simulation diagram for such a system is shown in Figure 3.6.

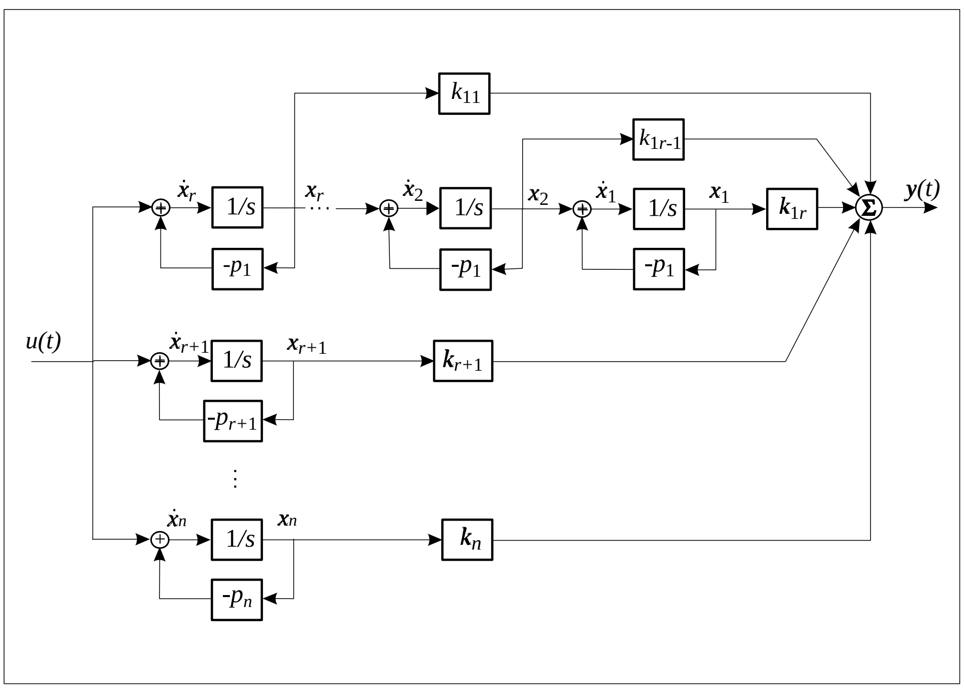

Figure 3.6: The simulation diagram for the Jordan canonical form

Taking for the state variables the outputs of integrators, the state space model is obtained as follows

$$\mathbf{A} = \begin{bmatrix} -p_1 & 1 & 0 & \cdots & \cdots & 0 & 0 & \cdots & \cdots & 0 \\ 0 & -p_1 & 1 & 0 & \cdots & 0 & 0 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & -p_1 & 1 & 0 & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & -p_1 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & -p_1 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & \cdots & \cdots & \cdots & 0 & -p_{r+1} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & \cdots & \cdots & 0 & \ddots & -p_{r+2} & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & \cdots & \cdots & 0 & 0 & \cdots & 0 & -p_n \end{bmatrix} \tag{3.26}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & \cdots & \cdots & 0 & 1 & 1 & \cdots & \cdots & 1 \end{bmatrix}^T$$

$$\mathbf{C} = \begin{bmatrix} k_{1r} & k_{1r-1} & \cdots & \cdots & k_{12} & k_{11} & k_{r+1} & k_{r+2} & \cdots & k_n \end{bmatrix}, \quad \mathbf{D} = 0$$

This form of the system model is known as the *Jordan canonical form*. The complete analysis of the Jordan canonical form requires a lot of space and time. However, understanding the Jordan form is crucial for correct interpretation of system stability, hence in the following chapter, the Jordan form will be completely explained.

Example 3.6: Find the state space model from the transfer function using the Jordan canonical form

$$G(s) = \frac{s^2 + 6s + 8}{(s+1)^2(s+3)}$$

This transfer function can be expanded as

$$G(s) = \frac{1.25}{s+1} + \frac{1.5}{(s+1)^2} - \frac{0.25}{s+3}$$

so that the required state space model is

$$\dot{\mathbf{x}} = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -3 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1.5 & 1.25 & -0.25 \end{bmatrix} \mathbf{x}$$

◇

3.4 The System Characteristic Equation and Eigenvalues

The characteristic equation is very important in the study of both linear time invariant continuous and discrete systems. No matter what model type is considered (ordinary nth-order differential equation, state space or transfer function), the characteristic equation always has the same form.

If we start with a differential nth-order system model in the operator form

$$\begin{aligned} &\left(p^n + a_{n-1}p^{n-1} + \cdots + a_1 p + a_0\right) y(t) \\ &\quad = \left(b_m p^m + b_{m-1} p^{m-1} + \cdots + b_1 p + b_0\right) u(t) \end{aligned}$$

where the operator p is defined as

$$p^i = \frac{d^i}{dt^i}, \quad i = 1, 2, \ldots, n-1$$

and $m \leq n$, then the *characteristic equation*, according to the mathematical theory of linear differential equations (Boyce and DiPrima, 1992), is defined by

$$s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0 = 0 \tag{3.27}$$

Note that the operator p is replaced by the complex variable s playing the role of a derivative in the Laplace transform context.

In the state space variable approach we have seen from (3.54) that

$$\mathbf{G}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} = \frac{1}{|s\mathbf{I} - \mathbf{A}|}\mathbf{C}[\mathrm{adj}(s\mathbf{I} - \mathbf{A})]\mathbf{B} + \mathbf{D}$$

$$= \frac{1}{|s\mathbf{I} - \mathbf{A}|}\{\mathbf{C}[\mathrm{adj}(s\mathbf{I} - \mathbf{A})]\mathbf{B} + |s\mathbf{I} - \mathbf{A}|\mathbf{D}\}$$

The characteristic equation here is defined by

$$|s\mathbf{I} - \mathbf{A}| = 0 \tag{3.28}$$

A third form of the characteristic equation is obtained in the context of the transfer function approach. The transfer function of a single-input single-output system is

$$G(s) = \frac{b_m s^m + b_{m-1}s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} \tag{3.29}$$

The characteristic equation in this case is obtained by equating the denominator of this expression to zero. Note that for multivariable systems, the characteristic polynomial (obtained from the corresponding characteristic equation) appears in denominators of all entries of the matrix transfer function.

No matter what form of the system model is considered, the characteristic equation is always the same. This is obvious from (3.95) and (3.97), but is not so clear from (3.96). It is left as an exercise to the reader to show that (3.95) and (3.96) are identical (Problem 3.30).

The *eigenvalues* are defined in linear algebra as scalars, λ, satisfying (Fraleigh and Beauregard, 1990)

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v} \tag{3.30}$$

where the vectors $\mathbf{v} \neq 0$ are called the *eigenvectors*. This system of n linear algebraic equations (λ is fixed) has a solution $\mathbf{v} \neq 0$ if and only if

$$|\lambda \mathbf{I} - \mathbf{A}| = 0 \tag{3.31}$$

Obviously, (3.96) and (3.99) have the same form. Since (3.96) = (3.95), it follows that the last equation is the characteristic equation, and hence the eigenvalues are the zeros of the characteristic equation. For the characteristic equation of order n, the number of eigenvalues is equal to n. Thus, the roots of the characteristic equation in the state space context are the eigenvalues of the matrix $\mathbf{A}$. These roots in the transfer function context are called the *system poles,* according to the mathematical tools for analysis of these systems—the complex variable methods.

Similarity Transformation

We have pointed out before that a system modeled by the state space technique may have many state space forms. Here, we establish a relationship among those state space forms by using a linear transformation known as the similarity transformation.

For a given system

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \quad \mathbf{x}(0) = \mathbf{x}_0$$
$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

we can introduce a new state vector $\hat{\mathbf{x}}$ by a linear coordinate transformation as follows

$$\mathbf{x} = \mathbf{P}\hat{\mathbf{x}}$$

where $\mathbf{P}$ is some nonsingular $n \times n$ matrix. A new state space model is obtained as

$$\begin{aligned} \dot{\hat{\mathbf{x}}} &= \hat{\mathbf{A}}\hat{\mathbf{x}} + \hat{\mathbf{B}}\mathbf{u}, \quad \hat{\mathbf{x}}(0) = \hat{\mathbf{x}}_0 \\ \mathbf{y} &= \hat{\mathbf{C}}\hat{\mathbf{x}} + \hat{\mathbf{D}}\mathbf{u} \end{aligned} \tag{3.32}$$

where

$$\hat{\mathbf{A}} = \mathbf{P}^{-1}\mathbf{A}\mathbf{P},\ \hat{\mathbf{B}} = \mathbf{P}^{-1}\mathbf{B},\ \hat{\mathbf{C}} = \mathbf{C}\mathbf{P},\ \ \hat{\mathbf{D}} = \mathbf{D},\ \hat{\mathbf{x}}(0) = \mathbf{P}^{-1}\mathbf{x}(0) \tag{3.33}$$

This transformation is known in the literature as the *similarity transformation*. It plays an important role in linear control system theory and practice.

Very important features of this transformation are that under similarity transformation both the system eigenvalues and the system transfer function are invariant.

Eigenvalue Invariance

A new state space model obtained by the similarity transformation does not change internal structure of the model, that is, the eigenvalues of the system remain the same. This can be shown as follows

$$\begin{aligned}\left|s\mathbf{I} - \hat{\mathbf{A}}\right| &= \left|s\mathbf{I} - \mathbf{P}^{-1}\mathbf{A}\mathbf{P}\right| = \left|\mathbf{P}^{-1}(s\mathbf{I} - \mathbf{A})\mathbf{P}\right| \\ &= \left|\mathbf{P}^{-1}\right|\left|s\mathbf{I} - \mathbf{A}\right|\left|\mathbf{P}\right| = \left|s\mathbf{I} - \mathbf{A}\right|\end{aligned} \tag{3.34}$$

Note that in this proof the following properties of the matrix determinant have been used

$$\det(\mathbf{M}_1\mathbf{M}_2\mathbf{M}_3) = \det\mathbf{M}_1 \times \det\mathbf{M}_2 \times \det\mathbf{M}_3$$

$$\det\mathbf{M}^{-1} = \frac{1}{\det\mathbf{M}}$$

see Appendix C.

Transfer Function Invariance

Another important feature of the similarity transformation is that the transfer function remains the same for both models, which can be shown as follows

$$\hat{\mathbf{G}}(s) = \hat{\mathbf{C}}\left(s\mathbf{I} - \hat{\mathbf{A}}\right)^{-1}\hat{\mathbf{B}} + \hat{\mathbf{D}} = \mathbf{CP}\left(s\mathbf{I} - \mathbf{P}^{-1}\mathbf{AP}\right)^{-1}\mathbf{P}^{-1}\mathbf{B} + \mathbf{D}$$
$$= \mathbf{CP}\left[\mathbf{P}^{-1}(\mathbf{sI} - \mathbf{A})\mathbf{P}\right]^{-1}\mathbf{P}^{-1}\mathbf{B} + \mathbf{D}$$
$$= \mathbf{CPP}^{-1}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{PP}^{-1}\mathbf{B} + \mathbf{D}$$
$$= \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} = \mathbf{G}(s) \tag{3.35}$$

Note that we have used in (3.103) the matrix inversion property (Appendix C)

$$(\mathbf{M}_1\mathbf{M}_2\mathbf{M}_3)^{-1} = \mathbf{M}_3^{-1}\mathbf{M}_2^{-1}\mathbf{M}_1^{-1}$$

The above result is quite logical—the system preserves its input–output behavior no matter how it is mathematically described.

Modal Transformation

One of the most interesting similarity transformations is the one that puts matrix $\mathbf{A}$ into diagonal form. Assume that $\mathbf{P} = \mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n]$, where $\mathbf{v}_i$ are the eigenvectors. We then have

$$\mathbf{V}^{-1}\mathbf{AV} = \hat{\mathbf{A}} = \Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n) \tag{3.36}$$

It is easy to show that the elements λ_i, $i = 1, ..., n$, on the matrix diagonal of Λ are the roots of the characteristic equation $|s\mathbf{I} - \Lambda| = |s\mathbf{I} - \mathbf{A}| = 0$, i.e. they are the eigenvalues. This can be shown in a straightforward way

$$|s\mathbf{I} - \Lambda| = \det\{diag(s - \lambda_1, s - \lambda_2, \ldots, s - \lambda_n)\}$$
$$= (s - \lambda_1)(s - \lambda_2)\cdots(s - \lambda_n)$$

The state transformation (3.104) is known as the *modal transformation*.

Note that the pure diagonal state space form defined in (3.104) can be obtained only in the following three cases.

1. The system matrix has distinct eigenvalues, namely $\lambda_1 \neq \lambda_2 \neq \cdots \neq \lambda_n$.
2. The system matrix is symmetric (see Appendix C).
3. The system minimal polynomial does not contain multiple eigenvalues. For the definition of the minimal polynomial and the corresponding pure diagonal Jordan form, see Subsection 4.2.4.

In the above three cases we say that the system matrix is diagonalizable.

Remark: Relation (3.104) may be represented in another form, that is

$$\mathbf{V}^{-1}\mathbf{A} = \Lambda\mathbf{V}^{-1}$$

or

$$\mathbf{W}^T\mathbf{A} = \Lambda\mathbf{W}^T$$

where

$$\mathbf{W}^T = \mathbf{V}^{-1} \ \Rightarrow \ \mathbf{W}^T\mathbf{V} = \mathbf{I}$$

In this case the *left eigenvectors* $\mathbf{w}_i, i = 1, 2, ..., n$, can be computed from

$$\mathbf{w}_i^T\mathbf{A} = \lambda_i\mathbf{w}_i^T \quad \Rightarrow \quad \mathbf{A}^T\mathbf{w}_i = \lambda_i\mathbf{w}_i$$

where $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n]$. Since $|\lambda\mathbf{I} - \mathbf{A}| = \left|\lambda\mathbf{I} - \mathbf{A}^T\right|$, then λ_i is also an eigenvalue of $\mathbf{A}^T$.

There are numerous program packages available to compute both the eigenvalues and eigenvectors of a matrix. In MATLAB this is done by using the function `eig`.

3.4.1 Multiple Eigenvalues

If the matrix $\mathbf{A}$ has multiple eigenvalues, it is possible to transform it into a block diagonal form by using the transformation

$$\mathbf{J} = \mathbf{V}^{-1}\mathbf{A}\mathbf{V} \tag{3.37}$$

where the matrix $\mathbf{V}$ is composed of n linearly independent, so-called *generalized eigenvectors* and $\mathbf{J}$ is known as the Jordan canonical form. This block diagonal form contains simple Jordan blocks on the diagonal. Simple Jordan blocks have the given eigenvalue on the main diagonal, ones above the main diagonal with all other elements equal to zero. For example, a simple Jordan block of order four is given by

$$\mathbf{J}_i(\lambda_i) = \begin{bmatrix} \lambda_i & 1 & 0 & 0 \\ 0 & \lambda_i & 1 & 0 \\ 0 & 0 & \lambda_i & 1 \\ 0 & 0 & 0 & \lambda_i \end{bmatrix}$$

Let the eigenvalue λ_1 have multiplicity of order 3 in addition to two real and distinct eigenvalues, $\lambda_2 \neq \lambda_3$; then a $\mathbf{J}$ matrix of order 5×5 may contain the following three simple Jordan blocks

$$\mathbf{J} = \begin{bmatrix} \lambda_1 & 1 & 0 & 0 & 0 \\ 0 & \lambda_1 & 1 & 0 & 0 \\ 0 & 0 & \lambda_1 & 0 & 0 \\ 0 & 0 & 0 & \lambda_2 & 0 \\ 0 & 0 & 0 & 0 & \lambda_3 \end{bmatrix}$$

However, other choices are also possible. For example, we may have the following distribution of simple Jordan blocks

$$\mathbf{J} = \begin{bmatrix} \lambda_1 & 1 & 0 & 0 & 0 \\ 0 & \lambda_1 & 0 & 0 & 0 \\ 0 & 0 & \lambda_1 & 0 & 0 \\ 0 & 0 & 0 & \lambda_2 & 0 \\ 0 & 0 & 0 & 0 & \lambda_3 \end{bmatrix} \quad \text{or} \quad \mathbf{J} = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 & 0 \\ 0 & \lambda_1 & 0 & 0 & 0 \\ 0 & 0 & \lambda_1 & 0 & 0 \\ 0 & 0 & 0 & \lambda_2 & 0 \\ 0 & 0 & 0 & 0 & \lambda_3 \end{bmatrix}$$

The study of the Jordan form is quite complex. Much more about the Jordan form will be presented in Chapter 4, where we study system stability.

3.4.2 Modal Decomposition

Diagonalization of matrix $\mathbf{A}$ using transformation $\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}$ makes the system $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$ diagonal, that is

$$\dot{\hat{\mathbf{x}}} = \Lambda\hat{\mathbf{x}} + \left(\mathbf{V}^{-1}\mathbf{B}\right)\mathbf{u} = \Lambda\hat{\mathbf{x}} + \left(\mathbf{W}^T\mathbf{B}\right)\mathbf{u}, \quad \hat{\mathbf{x}}(0) = \hat{\mathbf{x}}_0$$

In such a case the homogeneous equation $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x},\ \mathbf{x}(0) = \mathbf{x}_0$, becomes

$$\dot{\hat{\mathbf{x}}} = \Lambda\hat{\mathbf{x}}, \quad \hat{\mathbf{x}}(0) = \mathbf{V}^{-1}\mathbf{x}(0) = \mathbf{V}^{-1}\mathbf{x}_0$$

or

$$\dot{\hat{x}}_i = \lambda_i\hat{x}_i, \ \ i = 1, \ldots, n$$

This system is represented by n independent differential equations. The modal response to the initial condition is

$$\hat{\mathbf{x}}(t) = e^{\Lambda t}\hat{\mathbf{x}}_0 = e^{\Lambda t}\mathbf{V}^{-1}\mathbf{x}_0 = e^{\Lambda t}\mathbf{W}^T\mathbf{x}_0$$

or

$$\hat{x}_i(t) = \hat{x}_i(0)e^{\lambda_i t} = \left(\mathbf{w}_i^T\mathbf{x}_0\right)e^{\lambda_i t}$$

The response $\mathbf{x}(t)$ is a combination of the modal components

$$\mathbf{x}(t) = \mathbf{V}\hat{\mathbf{x}}(t) = \mathbf{V}e^{\Lambda t}\mathbf{V}^{-1}\mathbf{x}_0 = \mathbf{V}e^{\Lambda t}\mathbf{W}^T\mathbf{x}_0$$
$$= \left(\mathbf{w}_1^T\mathbf{x}_0\right)e^{\lambda_1 t}\mathbf{v}_1 + \left(\mathbf{w}_2^T\mathbf{x}_0\right)e^{\lambda_2 t}\mathbf{v}_2 + \cdots + \left(\mathbf{w}_n^T\mathbf{x}_0\right)e^{\lambda_n t}\mathbf{v}_n \tag{3.38}$$

This equation represents the modal decomposition of $\mathbf{x}(t)$ and it shows that the total response consists of a sum of responses of all individual modes. Note that $\mathbf{w}_i^T\mathbf{x}_0$ are scalars.

It is customary to call the reciprocals of λ_i the *system time constants* and denote them by τ_i, that is

$$\tau_i = \frac{1}{\lambda_i}, \quad i = 1, 2, ..., n$$

This has physical meaning since the system dynamics is determined by its time constants and these do appear in the system response in the form e^{-t/τ_i}.

The transient response of the system may be influenced differently by different modes, depending of the eigenvalues λ_i. Some modes may decay faster than the others. Some modes might be dominant in the system response. These cases will be illustrated in Chapter 6.

Remark: A similarity transformation $\Lambda = \mathbf{V}^{-1}\mathbf{A}\mathbf{V}$ can be used for the state transition matrix calculation. Recall

$$\hat{\mathbf{x}}(t) = e^{\Lambda t}\hat{\mathbf{x}}(0), \quad \hat{\mathbf{x}}(t) = \mathbf{V}\mathbf{x}(t), \quad \hat{\mathbf{x}}(0) = \mathbf{V}\mathbf{x}(0)$$

and

$$\mathbf{x}(t) = \mathbf{V}^{-1}e^{\Lambda t}\mathbf{V}\mathbf{x}(0) = \Phi(t)\mathbf{x}(0)$$

Hence,

$$\Phi(t) = e^{\Lambda t} = \mathbf{V}^{-1}e^{\Lambda t}\mathbf{V} = \mathbf{W}^T e^{\Lambda t}\mathbf{V}$$

or, in the complex domain

$$\Phi(s) = \mathbf{V}^{-1}(s\mathbf{I} - \Lambda)^{-1}\mathbf{V}$$
$$= \mathbf{V}^{-1} diag\{s - \lambda_1, s - \lambda_2, \ldots, s - \lambda_n\}^{-1}\mathbf{V}$$
$$= \mathbf{V}^{-1} diag\left\{\frac{1}{s - \lambda_1}, \frac{1}{s - \lambda_2}, \ldots, \frac{1}{s - \lambda_n}\right\}\mathbf{V}$$

Remark: The presented theory about the system characteristic equation, eigenvalues, eigenvectors, similarity and modal transformations can be applied directly to discrete-time linear systems with $\mathbf{A}_d$ replacing $\mathbf{A}$.