equations, giving n state variables. We then use the vector notation to collect together these n state variables as a state vector, $\mathbf{x}(t) = [x_1(t), x_2(t), \ldots , x_n(t)]^T$.

20.2.2 Matrices of coefficients

Often we find that the system variables are linked to the system inputs, outputs and other state variables through matrices of coefficients. Consider the following equation, where $\mathbf{x}(t) = [x_1(t), x_2(t), x_3(t)]^T$ is a function of variables $h(t)$, $m(t)$ and $p(t)$

$$x_1(t) = -3h(t) + 2m(t) + 6p(t)$$

$$x_2(t) = 4h(t) - 3m(t) + 9p(t)$$

$$x_3(t) = h(4) - p(t)$$

This can be written as

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} = \begin{bmatrix} -3 & 2 & 6 \\ 4 & -3 & 9 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} h(t) \\ m(t) \\ p(t) \end{bmatrix} = \begin{bmatrix} -3 & 2 & 6 \\ 4 & -3 & 9 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \\ u_3(t) \end{bmatrix}$$

which has the succinct form:

$$\mathbf{x}(t) = \begin{bmatrix} -3 & 2 & 6 \\ 4 & -3 & 9 \\ 1 & 0 & -1 \end{bmatrix} \mathbf{u}(t)$$

where we have let $\mathbf{u}(t) = [h(t), m(t), p(t)]^T$ and $\mathbf{x}(t) = [x_1(t), x_2(t), x_3(t)]^T$. This equation can also be written as

$$\mathbf{x}(t) = \mathbf{A}\mathbf{u}(t) \text{ where } \mathbf{A} = \begin{bmatrix} -3 & 2 & 6 \\ 4 & -3 & 9 \\ 1 & 0 & -1 \end{bmatrix}$$

We will use this type of vector–matrix notation whenever we meet state variable systems with more than one state.

20.3 General procedure for writing a state variable representation

We would like to form a state variable description of a general linear system (Figure 20.2). The system of Figure 20.2 can be represented in more detail by Figure 20.3.

We usually use the notation $\mathbf{u}(t)$ for inputs and $\mathbf{y}(t)$ for outputs, and we usually denote the state of the system by $\mathbf{x}(t)$. Up till now the systems analysed in this book have been represented by an *input–output* transfer function relationship between $Y(s)$ and $U(s)$. By representing the system in terms of the system states we not only have access to the inputs and outputs but we can produce a record of the internal variables – the system states – within the system. Previously we may have been restricted to knowledge of the measurable output, $y(t)$, but by forming a state variable model we can calculate the values of the measurable *and unmeasurable* system states. We find that in place of *output feedback*, which we have covered in this book up till now, we can move into the realm of *state feedback*, which can provide options which were not available previously. We investigate the control of a system

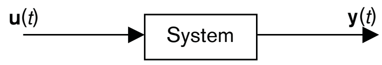


Figure 20.2 General system block diagram.

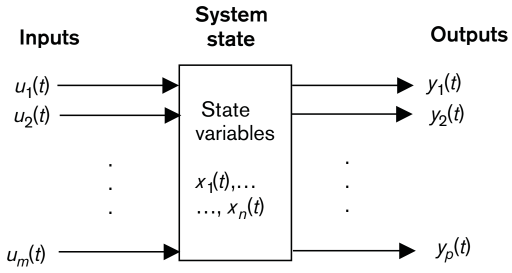


Figure 20.3 Inputs, outputs and state variables.

using state variables in Chapter 23. Another advantage of state variable notation is that this succinct notation gives us the means of writing a *multivariable* system description in a concise form. We have often met single-input single-output systems within this book, but the state variable notation, with its vector–matrix form is an immediate aid to writing multi-input multi-output system models.

We now provide a procedure for creating a state variable description of a system.

20.3.1 General procedure

(a) Define the system equations
We start by deriving the system equations, which are usually combinations of differential and algebraic equations. We consider ways in which they might be ordered in preparation for step (b).

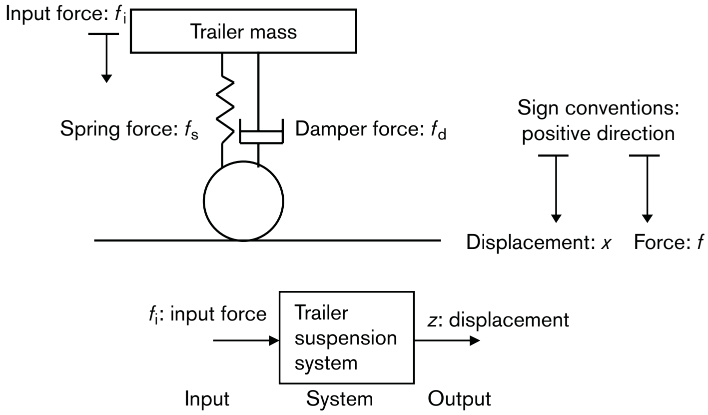
(b) Identify the system inputs, outputs and states
Since the state variable description relies on inputs, outputs and system states, we must define these for the system and reformulate the system equations using the new state variable notation

(c) Rewrite the new system equations in standard state variable vector–matrix notation
We introduce a systematic vector–matrix notation which uses four matrices: **A**, **B**, **C** and **D**. We collect together the equations to define the **A**, **B**, **C** and **D** matrices for the system.

We illustrate the procedure by applying it to the trailer suspension model of Chapter 7.

20.3.2 State variable model of trailer suspension system

(a) Define the system equations
We first examine the system and the equations which describe its dynamic behaviour. We remember that the model represents a second-order system where the position of the trailer mass is dependent on the input force, f_i (Figure 20.4).



System	Trailer suspension system
Differential equation	$\frac{M}{K_s}\frac{d^2z}{dt^2}+\frac{B}{K_s}\frac{dz}{dt}+z(t)=\frac{1}{K_s}f_i(t)$
Parameter K_s	80 000 N/m
Parameter B	3464 N/ms^{-1}
Parameter M	75 kg

Figure 20.4 Trailer suspension system and parameters.

The differential equation that describes the system is given by

$$\frac{M}{K_s}\frac{d^2z}{dt^2}+\frac{B}{K_s}\frac{dz}{dt}+z(t)=\frac{1}{K_s}f_i(t)$$

or

$$9.375\times10^{-4}\frac{d^2z}{dt^2}+4.55\times10^{-3}\frac{dz}{dt}+z(t)=1.25\times10^{-5}f_i(t)$$

(b) Identify the inputs, outputs and states

System inputs

We use the notation $\mathbf{u}(t)$ to represent the input signal to the system. In this example, there is one input signal, $f_i(t)$; therefore we define

$$\mathbf{u}(t) = [f_i(t)]$$

If there were more than one input signal, such as two forces $f_1(t)$ and $f_2(t)$, then the inputs would have been

$$u_1(t) = f_1(t)$$

$$u_2(t) = f_2(t)$$

and the input vector $\mathbf{u}(t)$ would have been given by

$$\mathbf{u}(t)=\begin{bmatrix}u_1(t)\\u_2(t)\end{bmatrix}=\begin{bmatrix}f_1(t)\\f_2(t)\end{bmatrix}$$

State variables

The system equation for the trailer suspension system is a second-order differential equation, and therefore we look for two states for the system, $x_1(t)$ and $x_2(t)$. We then aim to write two first-order differential equations for $\dot{x}_1(t)$ and $\dot{x}_2(t)$.

Let the first state be the position and the second state be the velocity

$$x_1(t) = z(t) \qquad \text{(position)}$$

$$x_2(t) = \frac{dz}{dt} \qquad \text{(velocity)}$$

From the definition of velocity we find that the first differential equation is given by

$$\dot{x}_1(t) = \frac{dz}{dt} = x_2(t).$$

For the second differential equation, we use the equation in terms of the highest derivative and rewrite it in the form

$$\frac{d^2z}{dt^2} = -\frac{B}{M}\frac{dz}{dt} - \frac{K_s}{M}z(t) + \frac{1}{M}f_i(t)$$

We now substitute in this equation the state variable notation:

$$x_1(t) = z(t) \qquad \text{(position)}$$

$$x_2(t) = \frac{dz}{dt} \qquad \text{(velocity)}$$

$$\dot{x}_2(t) = \frac{d^2z}{dt^2} \qquad \text{is the derivative of the highest state (acceleration)}$$

Input signal: $\mathbf{u}(t) = f_i(t)$

Substituting these variables gives

$$\dot{x}_2(t) = -\frac{B}{M}x_2(t) - \frac{K_s}{M}x_1(t) + \frac{1}{M}\mathbf{u}(t)$$

This is a first-order differential equation for $\dot{x}_2(t)$. We also use the first-order equation relating states 1 and 2 (position and velocity) for the other first-order differential equation in $x_1(t)$.

Summary of state equations

$$\dot{x}_1(t) = x_2(t)$$

$$\dot{x}_2(t) = -\frac{B}{M}x_2(t) - \frac{K_s}{M}x_1(t) + \frac{1}{M}\mathbf{u}(t)$$

System outputs

We use the notation $\mathbf{y}(t)$ to represent the system's output signal. We have only one output signal, the position $z(t)$, which, in this example, is also the first state variable, $x_1(t)$:

$$\mathbf{y}(t) = z(t) = x_1(t)$$

Note: if there were more than one output signal, such as position *and* velocity ($x_1(t)$ and $x_2(t)$), then the outputs would have been

$$y_1(t) = x_1(t)$$

$$y_2(t) = x_2(t)$$

and the output vector $\mathbf{y}(t)$ would be given by

$$\mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}$$

(c) Vector–matrix ABCD notation

Although we have written the second-order differential equation in terms of two first-order equations, the notation is still not very concise. We adopt a matrix convention by rewriting the state variable equations in vector–matrix form, followed by the output equation.

We have two first-order equations replacing the second-order differential equation

$$\dot{x}_1(t) = x_2(t)$$

$$\dot{x}_2(t) = -\frac{B}{M}x_2(t) - \frac{K_s}{M}x_1(t) + \frac{1}{M}\mathbf{u}(t)$$

We define the state vector

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

with

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix}$$

and go to a vector–matrix form directly from the above equations:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -K_s/M & -B/M \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1/M \end{bmatrix} \mathbf{u}(t)$$

or

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 1 \\ -K_s/M & -B/M \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ 1/M \end{bmatrix} \mathbf{u}(t)$$

This can be written as

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

where

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -K_s/M & -B/M \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 \\ 1/M \end{bmatrix}$$

The output equation is given by

$$\mathbf{y}(t) = x_1(t)$$

and this can be written in terms of the state vector $\mathbf{x}(\mathrm{t}) = [x_1(t), x_2(t)]^T$ and the control input $\mathbf{u}(t)$ as

$$\mathbf{y}(t) = [1 \quad 0]\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = [1 \quad 0]\mathbf{x}(t)$$

However, for completeness we can write the output equation as a function of both the state $\mathbf{x}(t)$ and the input $\mathbf{u}(t)$: $\mathbf{y}(\mathbf{x}(t), \mathbf{u}(t))$. In this example, we have

$$\mathbf{y}(t) = [1 \quad 0]\mathbf{x}(t) + [0]\mathbf{u}(t)$$

This leads us to the form

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

where $\mathbf{C} = [1, 0]$ and $\mathbf{D} = [0]$.

We summarise the complete state variable system model description in the following.

Key result: State variable notation summary

Given a system of

m inputs
n states
r outputs

the full state space system is given by

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

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

where **A** (size $n \times n$) is the system matrix
B (size $n \times m$) is the input matrix
C (size $r \times n$) is the output matrix
D (size $r \times m$) is the direct feedthrough matrix

The matrix **D** represents any direct connections between the input and the output. However, in many simple cases, such as the trailer suspension example, the **D** matrix is zero.

Skill section — Writing vector–matrix forms for state variable ABCD matrices

The rewriting of state variable equations in vector–matrix form will occur often in state space work. The step of identifying the number of states (*n*), inputs (*m*) and outputs (*r*) automatically sets up the size of the **ABCD** matrices to be filled:

$$\dot{\mathbf{x}}(t) = \mathbf{A}_{n\times n}\mathbf{x}(t) + \mathbf{B}_{n\times m}\mathbf{u}(t)$$
$$\mathbf{y}(t) = \mathbf{C}_{r\times n}\mathbf{x}(t) + \mathbf{D}_{r\times m}\mathbf{u}(t)$$

Problem For the following two systems, determine the dimensions of the **A**, **B**, **C** and **D** matrices.

System 1: Number of states is 3, number of inputs is 2, number of outputs is 1.
System 2: Number of states is 6, number of inputs is 6, number of outputs is 3.

Solution System 1:

We are given the information: $n = 3$, $m = 2$ and $r = 1$. Therefore

A: (3 × 3); **B**: (3 × 2); **C**: (1 × 3); and **D**: (1 × 2)

System 2:
We are given the information: $n = 6$, $m = 6$ and $r = 3$. Therefore

A: (6 × 6); **B**: (6 × 6); **C**: (3 × 6); and **D**: (3 × 6)

20.4 State variable diagram

We can represent the state variable model diagrammatically as in Figure 20.5. We have often used block diagrams to represent control systems. The state variable diagram has a particular format. This started from the time when digital computers were not available and analogue circuits were used to form integrating components and the gain blocks.

The forms of Figure 20.5 have been used to represent an integral component.

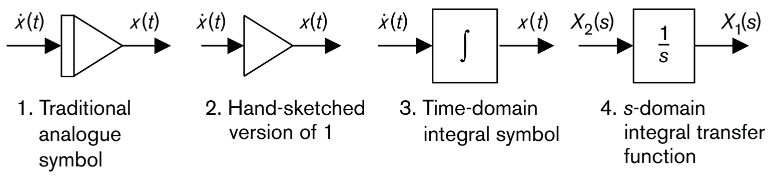


Figure 20.5 State variable block representations.

We will use blocks 3 and 4, depending on whether we are using the time domain or the s-domain in our block diagram. In the time domain, the general state variable diagram then looks like Figure 20.6. We have shown the lines connecting the **D** matrix as dotted, since for many examples in this book the **D** matrix will be zero, and these connections will not be present.

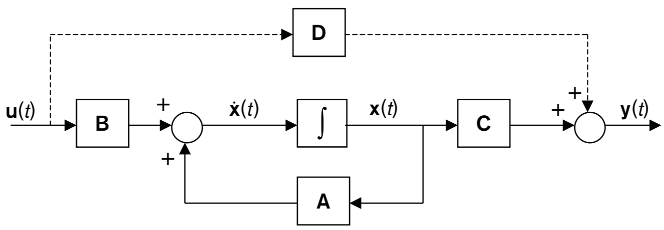


Figure 20.6 General ABCD block diagram.

The ABCD model represents a set of first-order differential equations which can be integrated to find the values of the system states. We require a set of initial conditions, $\mathbf{x}(0)$, to be able to solve the differential equations exactly. Since there are n differential equations we have an n-dimensional vector containing the n initial conditions. Very often these initial conditions are zero and we do not put them in our diagram. However, to be fully correct, we would add $\mathbf{x}(0)$ to the diagram of Figure 20.6 to give Figure 20.7.

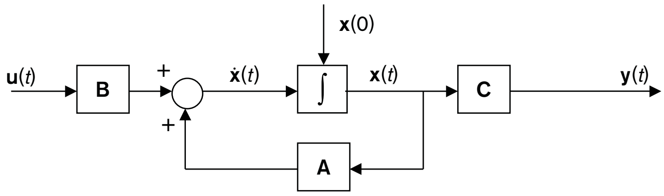
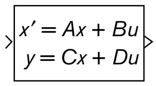


Figure 20.7 Initial conditions in a state variable system.

20.5 MATLAB–Simulink representation of state variable models

In MATLAB there is a block, called state-space (under Simulink, Continuous, Statespace), that represents a state space or state variable model.



State space

We can see that the block icon shows the state variable equations. It has one input port for $\mathbf{u}(t)$ and one exit port for $\mathbf{y}(t)$.

MATLAB requires all four matrices to be entered into the system. Even if there is no **D** matrix in the model, MATLAB will require a matrix of zeros to be entered. The **D** matrix should be the size [number of outputs × number of inputs].

Example: Trailer suspension model

We wish to enter the following trailer suspension model into a Simulink format:

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \qquad \mathbf{y}(t) = [y(t)] \qquad \mathbf{u}(t) = [u(t)]$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -K_s/M & -B/M \end{bmatrix} \qquad \mathbf{B} = [0 \quad 1/M] \qquad \mathbf{C} = [1 \quad 0]$$

where $K_s = 80\,000$ N/m, $B = 3464$ N/m s^{-1} and $M = 75$ kg.

Since we intend using an ABCD model, we must first ensure that we know the size and values of all four matrices. The **D** matrix often has zero entries, but we must check the size of the matrix. In this example, the **D** matrix will be zero and the size is given by $r \times m$ where r and m represent the number of outputs and number of inputs respectively. The trailer suspension example has only one input and one output, giving a size for **D** of 1 × 1.

We can enter now this model into a Simulink ABCD block easily. However, by using features of MATLAB we can make life easier for ourselves.

1. *Direct entry of parameter values*

The most obvious way to enter the matrices is by double-clicking on the ABCD-Simulink block, which reveals the data entry table (Figure 20.8).

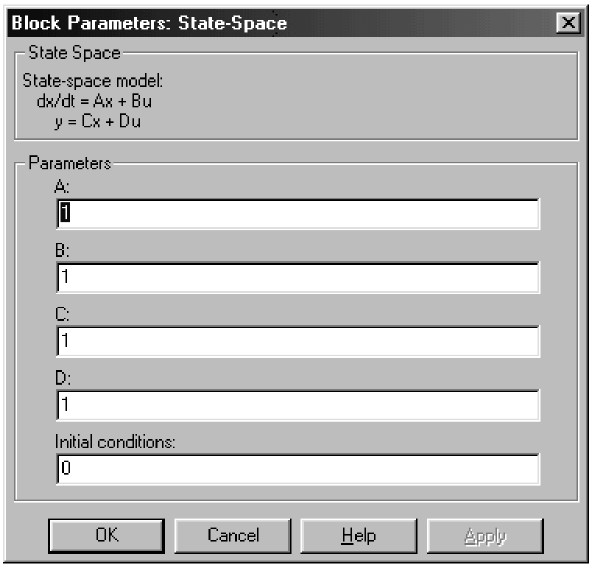


Figure 20.8 Data entry table for Simulink ABCD block.

We can then enter the matrices directly as:

A: [0 1; −1066.7 −46.187]

B: [0; 0.01333]

C: [1 0]

D: [0]

2. *Direct entry of parameter calculations*
We could have defined variables `Ks`, `M` and `B` in the MATLAB workspace, and entered the data as

A: `[0 1; -Ks/M -B/M]`

B: `[0; 1/M]`

C: `[1 0]`

D: `[0 ]`

3. *Entry of matrix names*
The Simulink data entry table could contain simply the matrix notation `A`, `B`, `C` and `D` in the corresponding entry places and we could could enter the parameter data and calculations in a MATLAB M-file which we would call before running the simulation. This is in most cases preferable, since we do not need to retype data and any changes to the data are made in one place and stored on file.

The M-file would contain

```
Ks = 80 000;      % Stiffness value in N/m
B = 3464;         % Damping value in N/ms–1
M = 75;           % Mass in kg
A = [0 1; -Ks/M -B/M]; B=[0; 1/M];
C = [1 0]; D=[0];
```

We tend to develop and use state variable descriptions in their physical parameter form rather than insert the numerical data. This allows us to follow the development of the equations more clearly, and only at the final step will we substitute the parameter values in to the equations. As shown above, it is easy to use data files containing parameter values and subsequent calculations rather than to enter data directly. This allows us the flexibility to change parameters more easily in one data file rather than throughout the simulation files, as well as having a file which records the values used.

In the next problem, taken from electrical engineering, the modelling equations have been given to us; this enables even the non-electrical engineer to complete the exercise.

20.6 Example of the development of a state variable model

Problem The following set of electrical equations describe the behaviour of the currents and voltages in the electrical circuit in Figure 20.9.

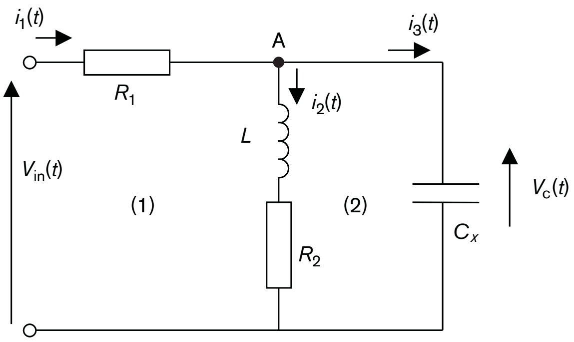


Figure 20.9 *RLC* circuit.

System equations

The table provides the system equations for the circuit.

Voltage round loop (1)	$V_{in}(t) = i_1(t)R_1 + L\dfrac{di_2(t)}{dt} + i_2(t)R_2$
Voltage across the capacitor	$C_x\dfrac{dV_c(t)}{dt} = i_3(t)$
Voltage round combined loops	$V_{in}(t) = i_1(t)R_1 + V_c(t)$
Current at node A	$i_1(t) = i_2(t) + i_3(t)$

Let the input $u(t)$ be the applied voltage, $V_{in}(t)$, the output signal be the voltage across the capacitor, C_x and the system states be given by

$x_1(t) = i_2(t)$: current through the inductor

$x_2(t) = V_c(t)$: voltage across the capacitor

Produce a state variable ABCD representation for this system.