Parasuram
July 13, 2001

HANDOUT A.3 - FORMULATING THE STATE-VARIABLE FORM OF DYNAMIC SYSTEMS

Introduction

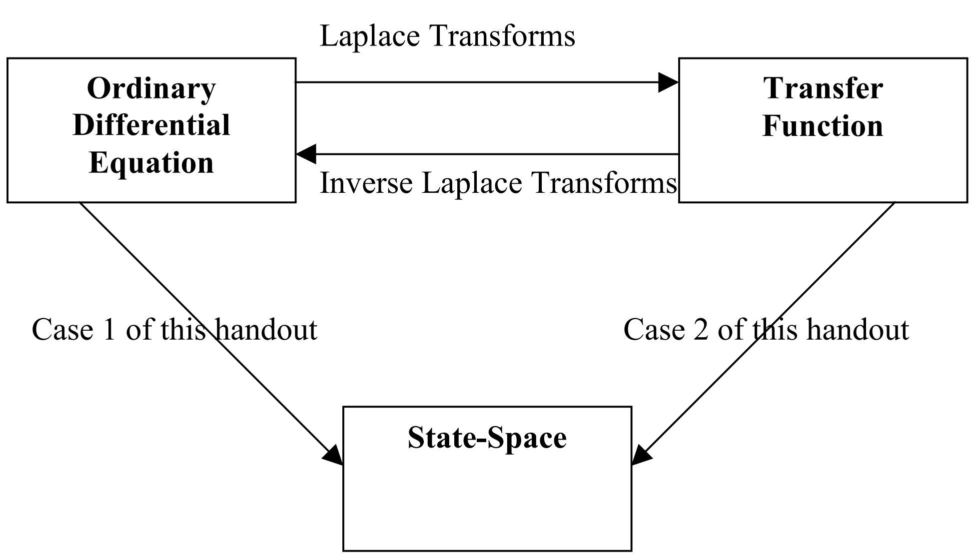
Use of Newton's law and free-body diagram typically leads to second order differential equations, that is, equations that contain the second derivative such as $\ddot{x}$. The differential equations can be expressed as a set of simultaneous first order differential equations. These are represented in the state-variable form as a vector equation

$$
\begin{aligned}
\dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \\
y(t) &= \mathbf{C}\mathbf{x}(t) + Du(t)
\end{aligned}
\qquad (1)
$$

where, for an nth order system, **A** is an n x n *system matrix*, **B** is an n x 1 *input matrix*, **C** is a 1 x n row matrix referred to as the *output matrix* and **D** is a scalar called the *direct transmission term*. The '**A**', '**B**', '**C**' and '**D**' matrices together define the system.

The column vector '**x**' is called the state of the system. ***The state of the system is a set of variables such that the knowledge of these variables and the input functions will, with the equations describing the dynamics, provide the future state and the output of the system***. In other words, the state variables describe the future response of a system, given the present state, the excitation inputs and the equations describing the dynamics. For mechanical systems, the state vector elements usually consist of the positions and velocities of the separate bodies.

The ordinary differential equations (ODE), which are obtained from the laws of physics, not only can be converted to state-space form but also to transfer functions. This handout explains the procedure to be followed to convert the ODE's to state-space form. The following diagram explains in detail the relationships between the various representations of a physical system.



Case 1: Formulation of the state-space equation from ODE

In general an nth order differential equation is given as

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

To reduce the above differential equation into a system of first order differential equations, let

$$y(t) = x_1;$$

$$\frac{dy(t)}{dt} = x_2;$$

$$\frac{d^2 y(t)}{dt^2} = x_3;$$

•

•

•

$$\frac{d^{n-1} y(t)}{dt^{n-1}} = x_n.$$

From the first two equations we have

$$\dot{x}_1 = \frac{dy}{dt} = x_2$$

which is a first order differential equation. Note that in the above equation, the dependence of 't' is implied and hence dropped from the equation.

Similarly from the second and the third equation, we get a first order differential equation:

$$\dot{x}_2 = \frac{d}{dt}\left(\frac{dy}{dt}\right) = \frac{d^2 y}{dt^2} = x_3$$

By following the above procedure for the next two equations, we get another first order differential equation. In this manner, we get (n-1) first order differential equations. Now substituting the above relations in equation (2) we have

$$\frac{d^n y}{dt^n} + a_1 \frac{d^{n-1} y}{dt^{n-1}} + \cdots + a_{n-1} \frac{dy}{dt} + a_n y = F$$

$$\Rightarrow \dot{x}_n + a_1 x_n + \cdots + a_{n-1} x_2 + a_n x_1 = F$$

$$\Rightarrow \dot{x}_n = F - a_1 x_n - \cdots - a_{n-1} x_2 - a_n x_1$$

From the above equations, it can be concluded that the nth order differential equation has been reduced to 'n' first order differential equation. Representing the first order differential equation in the form of a matrix, we have

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \cdot \\ \cdot \\ \cdot \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & \cdot & \cdot & \cdot & 0 \\ 0 & 0 & 1 & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ -a_n & -a_{n-1} & \cdot & \cdot & \cdot & -a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix} F$$

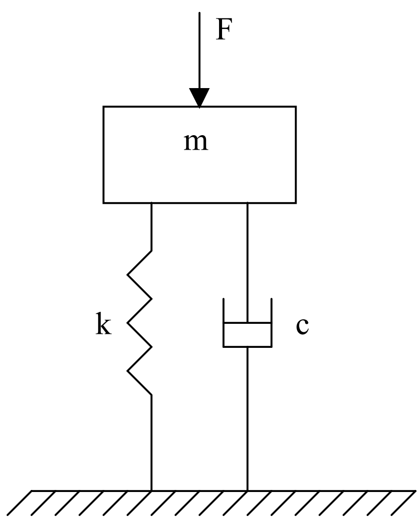
which is similar to the first equation of equation (1), which is

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

This is further explained with a help of a simple example.

Example 1

Consider the mass damper system shown below. The objective of this example is to write the governing differential equation in the state-variable form.



The governing differential equation of motion for the above system is given as

$$m\ddot{y} + c\dot{y} + ky = F \tag{3}$$

It can be seen that the above differential equation is a second order one. To reduce it to first order differential equation

Let

$$y = x_1$$
$$\frac{dy}{dt} = \dot{y} = x_2 \qquad (4)$$

So in this case the states of the system are x_1 and x_2, which are equal to y and $\dot{y}$ respectively. From the above two relations, it can be concluded that

$$\dot{x}_1 = \frac{dy}{dt} = x_2 \qquad (5)$$

which is a first order differential equation.

Substituting the relations (4) in equation (3) we get the second first order differential equation,

$$m\frac{d^2y}{dt^2} + c\frac{dy}{dt} + ky = F$$
$$\Rightarrow m\dot{x}_2 + cx_2 + kx_1 = F \qquad (6)$$
$$\Rightarrow \dot{x}_2 = \frac{F}{m} - \frac{c}{m}x_2 - \frac{k}{m}x_1$$

Writing equation (5) and the last expression of equation (6) in the form of a matrix, we have

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} F \qquad (7)$$

If for example, we are interested in measuring the velocity of the mass, then the output of the system is the velocity of the mass. In other words,

$$y = x_2 = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad (8)$$

Equations (7) and (8), which are similar to equations (1) represent the state variable form of the above-defined system. 'F' is the input to the system, 'x_1' and 'x_2', which denote the position and velocity of the mass represent the states of the system, and 'y' is the output of the system.

Comparing equations (7) and (8) with equations (1), the state matrix **A**, input matrix **B** and the output matrix **C** are given as

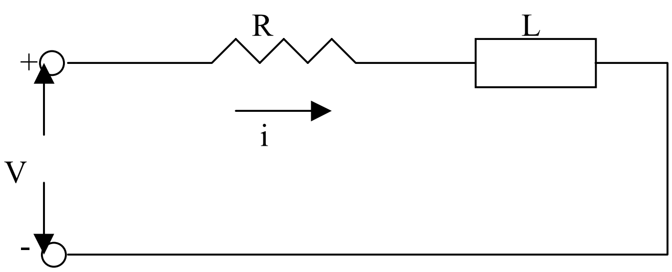
$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix};$$

$$\mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix};$$

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

Example 2

Consider the electrical circuit shown below. We want to represent the governing differential equation in state-variable form, where one of the states is the electric charge q(t). Assume that the output voltage is the voltage across the resistance.



The governing differential equation is given by

$$V = L\frac{di}{dt} + Ri \tag{9}$$

We know that

$$i = \frac{dq}{dt} \tag{10}$$

Substituting equation (10) in equation (9), we have

$$V = L\frac{d^2q}{dt^2} + R\frac{dq}{dt}$$

It can be clearly seen that the above equation is second order differential equation.

To reduce it to a state variable form, let the states of the system be defined as

$$q = x_1;$$
$$\frac{dq}{dt} = x_{2;} \tag{11}$$

From the above result, the first reduced order differential equation is

$$\dot{x}_1 = x_2 \tag{12}$$

Substituting the relations (11) in equation (9), we get the second reduced order differential equation as

$$V = L\dot{x}_2 + Rx_2$$
$$\Rightarrow \dot{x}_2 = \frac{V}{L} - \frac{R}{L}x_2 \tag{13}$$

Representing equations (12) and (13) in the state variable form, we get

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \frac{V}{L} \tag{14}$$

Since the output is the voltage across the resistance, we have

$$y = V_R = R\,i = R\frac{dq}{dt} = Rx_2$$

Representing the above equation in the matrix form, we get

$$y = \begin{bmatrix} 0 & R \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{15}$$

Equations (14) and (15) represent the state–space of the defined electrical system. The '**A**', '**B**', '**C**' and 'D' matrices can be obtained by comparing equations (14) and (15) to equation (1).

Case 2: Canonical Forms- Formation of state-space equation from transfer function

There are various ways in which the '**A**', '**B**', '**C**', '**D**' matrices can be represented. These various ways of representation are called the canonical form of the state-space. One such form is the *control canonical form.* Consider the transfer function

$$F(s) = \frac{P(s)}{Q(s)} = \frac{b_1 s^{n-1} + b_2 s^{n-2} + \cdots + b_n}{s^n + a_1 s^{n-1} + \cdots + a_n} \tag{16}$$

Then the control canonical form is

$$\mathbf{A} = \begin{bmatrix} -a_1 & -a_2 & \cdot & \cdot & \cdot & -a_n \\ 1 & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 1 & 0 & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & \cdot & 1 & 0 \end{bmatrix}; \quad \mathbf{B} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \cdot \\ \cdot \\ 0 \end{bmatrix}$$

$$\mathbf{C} = [b_1 \quad b_2 \quad \cdot \quad \cdot \quad \cdot \quad b_n]; \quad D = 0$$

So once the transfer function is obtained, the control canonical form can be easily written down based on the coefficients in the numerator and the denominator of the transfer function.

The other canonical forms are

- Observable canonical form
- Jordan canonical form
- Modal canonical form
- Diagonal canonical form

Example 3

Write down the state-space matrices based on the given transfer function.

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

Comparing the above given transfer function to equation (16), we get

$b_1 = 2;$
$b_2 = 3;$
$b_3 = 1;$
$a_1 = 4;$
$a_2 = 6;$
$a_3 = 5;$

Based on the values of the a's and the b's, the system matrices can be written as

$$\mathbf{A} = \begin{bmatrix} -4 & -6 & -5 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}; \quad \mathbf{B} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}; \quad \mathbf{C} = \begin{bmatrix} 2 & 3 & 1 \end{bmatrix}$$

Using MATLAB to perform various state-space operations

1) To create a state-space system, given the state, input and the output matrices, use the '*ss*' command in MATLAB. The online help in MATLAB gives the following:

```
help ss

 SS  Create state-space model or convert LTI model to state-space.

   Creation:
     SYS = SS(A,B,C,D) creates a continuous-time state-space (SS) model
     SYS with matrices A,B,C,D.  The output SYS is a SS object.  You
     can set D=0 to mean the zero matrix of appropriate dimensions.

     SYS = SS(A,B,C,D,Ts) creates a discrete-time SS model with sample
     time Ts (set Ts=-1 if the sample time is undetermined).

     SYS = SS creates an empty SS object.
     SYS = SS(D) specifies a static gain matrix D.

     In all syntax above, the input list can be followed by pairs
        'PropertyName1', PropertyValue1, ...
     that set the various properties of SS models (type LTIPROPS for
     details).  To make SYS inherit all its LTI properties from an
           existing   LTI   model   REFSYS,   use   the   syntax   SYS   =
      SS(A,B,C,D,REFSYS).

   Arrays of state-space models:
     You can create arrays of state-space models by using ND arrays for
     A,B,C,D above.  The first two dimensions of A,B,C,D determine the
     number of states, inputs, and outputs, while the remaining
     dimensions specify the array sizes.  For example, if A,B,C,D are
     4D arrays and their last two dimensions have lengths 2 and 5, then
        SYS = SS(A,B,C,D)
     creates the 2-by-5 array of SS models
```

```
       SYS(:,:,k,m) = SS(A(:,:,k,m),...,D(:,:,k,m)),  k=1:2,  m=1:5.
    All models in the resulting SS array share the same number of
    outputs, inputs, and states.

          SYS = SS(ZEROS([NY NU S1...Sk])) pre-allocates space for an
     SS array
    with NY outputs, NU inputs, and array sizes [S1...Sk].

  Conversion:
    SYS = SS(SYS) converts an arbitrary LTI model SYS to state-space,
    i.e., computes a state-space realization of SYS.

    SYS = SS(SYS,'min') computes a minimal realization of SYS.
```

Given a state-space system, the system matrices can be extracted by using the '*ssdata*' command in MATLAB. To know more about the command, use the '*help*' command.

```
help ssdata

 --- help for ss/ssdata.m ---

 SSDATA  Quick access to state-space data.

    [A,B,C,D] = SSDATA(SYS) retrieves the matrix data A,B,C,D
    for the state-space model SYS.  If SYS is not a state-space
    model, it is first converted to the state-space representation.

    [A,B,C,D,TS] = SSDATA(SYS) also returns the sample time TS.
    Other properties of SYS can be accessed with GET or by direct
    structure-like referencing (e.g., SYS.Ts).

    For arrays of LTI models with the same order (number of states),
    A,B,C,D are multi-dimensional arrays where A(:,:,k), B(:,:,k),
    C(:,:,k), D(:,:,k) give the state-space matrices of the
    k-th model SYS(:,:,k).

    For arrays of LTI models with variable order, use the syntax
       [A,B,C,D] = SSDATA(SYS,'cell')
    to return the variable-size A,B,C matrices into cell arrays.
```

The use of the above-mentioned commands is explained with the help of an example.

Example 3

Let the state, input and the output matrices be defined as

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}; \quad \mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \quad \mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}; \quad D = 0$$

The MATLAB code is as given below

```
A=[1 2; 3 4];
B=[0;1];
C=[1 0];
D=0;
sys=ss(A,B,C,D)
```

The first four lines of the above code represents the model defined in the example. They are entered in MATLAB in the form of matrices. The result of the above code is as shown below

```
a =
                         x1            x2
           x1             1             2
           x2             3             4


b =
                         u1
           x1             0
           x2             1


c =
                         x1            x2
           y1             1             0


d =
                         u1
           y1             0

Continuous-time model.
```

Suppose that the system 'sys' is given, as before, then the state-space matrices must be extracted, using the following code for further manipulation

```
[A,B,C,D]=ssdata(sys)
```

The result of the above code is

```
A =

     1     2
     3     4


B =

     0
     1
```

```
C =

     1     0

D =

     0
```

Note that, the state-space matrices are obtained back, using the '*ssdata*' command.

2) To convert a transfer function to state-space, use the '*tf2ss*' command, while to convert the state-space to transfer function, use the '*ss2tf*' command in MATLAB. This is further explained with the help of an example.

Example 4

Let the transfer function of a system be defined as

$$F(s) = \frac{1}{s^2 + 2s + 1}$$

Find the state-space form of the above transfer function.

The MATLAB code is

```
numerator = 1;
denominator = [1 2 0];
[A,B,C,D]=tf2ss(numerator, denominator)
```

The result is

```
A =

    -2     0
     1     0


B =

     1
     0


C =

     0     1


D =

     0
```

The online help can be used to get more information about the commands. Note that in the code, the numerator and the denominator of the transfer function is given in the form of a matrix. The elements of the matrix are the coefficients of the respective polynomials.

Assignment

1) For the state-space equation given below, determine the system transfer function.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

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

2) For the system transfer function defined by

$$Y(s) = \frac{(s+2)}{(s^3 + 3s^2 + 4s + 6)}$$

Determine the state, input and the output matrices using MATLAB and also represent the matrices in the control canonical form.

Recommended reading

"Feedback Control of Dynamic Systems" 4th Edition, by Gene F. Franklin et.al – pp 41-45, 494-509.