Christopher Lum
lum@uw.edu

Lecture 03b
State Space to Transfer Function

The YouTube video entitled 'State Space to Transfer Function' that covers this lecture is located at https://youtu.be/NNJ0sUmrKu8.

Outline

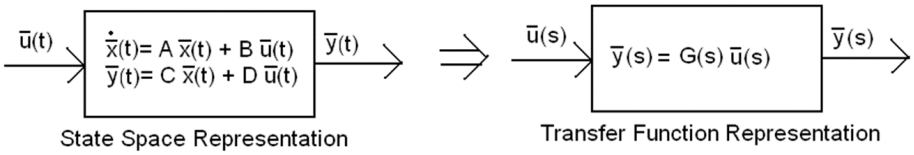
-Introduction
-State Space to Transfer Function
-Matlab Tools

Introduction

We know that there are several representations of a dynamic system. It would be useful to be able to translate between them.

State Space To Transfer Function

If you require a refresher on state space representations, please see the video entitled 'State Space Representation of Differential Equations' at https://youtu.be/pXvAh1IOO4U.



Recall that the state trajectory is governed by

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

Taking the Laplace transform of both side yields

$s\,\overline{X}(s) - \overline{x}(0) = A\,\overline{X}(s) + B\,\overline{U}(s)$ recall: for a transfer function, we assume no initial conditions $\overline{x}(0) = \overline{0}$

$$s\,\overline{X}(s) - A\,\overline{X}(s) = B\,\overline{U}(s)$$

$$(s\,I - A)\,\overline{X}(s) = B\,\overline{U}(s)$$

$$(s\,I - A)^{-1}\,(s\,I - A)\,\overline{X}(s) = (s\,I - A)^{-1}\,B\,\overline{U}(s)$$

$$\overline{X}(s) = (s\,I - A)^{-1}\,B\,\overline{U}(s) \qquad \textbf{(Eq.1)}$$

Recall that the output of the system is given by

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

Taking the Laplace transform of both sides yields

$\overline{Y}(s) = C\,\overline{X}(s) + D\,\overline{U}(s)$ recall: $\overline{X}(s) = (s\,I - A)^{-1}\,B\,\overline{U}(s)$

$$= C\,(s\,I - A)^{-1}\,B\,\overline{U}(s) + D\,\overline{U}(s)$$

$$= \left(C(s\,I - A)^{-1}\,B\, + D\right)\overline{U}(s)$$

$$\overline{Y}(s) = G(s)\,\overline{U}(s) \qquad \textbf{(Eq.2)}$$

where $G(s) = C(s\,I - A)^{-1}\,B\, + D$

Notice that now $G(s)$ is a matrix. For example, if your system has m outputs an p inputs, then $G(s)$ is an m – by – p sized matrix.

Example 1: DC Motor

Recall the from our discussion on modeling a DC motor as a dynamic system, we derived a state space representation of the system with the state and control vector of

$$\overline{x}(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{pmatrix} = \begin{pmatrix} \theta(t) \\ \dot{\theta}(t) \\ i(t) \end{pmatrix}$$

$$\overline{u}(t) = \begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \begin{pmatrix} V_a(t) \\ T_L(t) \end{pmatrix}$$

We assume sensors of

$$\overline{y}(t) = \begin{pmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{pmatrix} = \begin{pmatrix} \dot{\theta}(t) \\ \theta(t) \\ T(t) \end{pmatrix}$$ (note the first output is the velocity, second is the angle)

Recall the state space representation of the DC motor with was

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

where $A = \begin{pmatrix} 0 & 1 & 0 \\ 0 & -\frac{c}{J_m} & \frac{K_T}{J_m} \\ 0 & -\frac{K_V}{L_a} & -\frac{R_m+R}{L_a} \end{pmatrix}$ $B = \begin{pmatrix} 0 & 0 \\ 0 & -\frac{1}{J_m} \\ \frac{1}{L_a} & 0 \end{pmatrix}$

$C = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & K_T \end{pmatrix}$ $D = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$

Where the constants are defined as

```
%% Define constants
R   = 0.05;          %resistance of measurement resistor (ohms)
KV  = 0.09854;       %electrical machine constant (volt sec)
KT  = 0.09854;       %mechanical machine constant (Nm/amp)
Rm  = 1.5398;        %resistance of motor (ohms)
La  = 0.0015581;     %inductance of motor (henries)
c   = 0.00039719;    %coefficient of viscous friction (Nm sec)
Jm  = 0.00137;       %moment of inertia about axis of rotation (kg*m^2)
```

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & -c/Jm & KT/Jm \\ 0 & -KV/La & -(Rm+R)/La \end{pmatrix};\ \mathbf{B} = \begin{pmatrix} 0 & 0 \\ 0 & -1/Jm \\ 1/La & 0 \end{pmatrix};$$

$$\mathbf{Cmat} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & KT \end{pmatrix};$$

$$\mathbf{Dmat} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix};$$

To compute the transfer function representation, we can use the equation $G(s) = C(s\,I - A)^{-1}\,B + D$

```
G[s_] = Cmat.Inverse[s * IdentityMatrix[3] - A].B + Dmat // Simplify;
G[s] // MatrixForm
```

$$\begin{pmatrix} \frac{KT}{KT\,KV + (c+Jm\,s)(R+Rm+La\,s)} & -\frac{R+Rm+La\,s}{KT\,KV + (c+Jm\,s)(R+Rm+La\,s)} \\ \frac{KT}{s\,(KT\,KV + (c+Jm\,s)(R+Rm+La\,s))} & -\frac{R+Rm+La\,s}{s\,(KT\,KV + (c+Jm\,s)(R+Rm+La\,s))} \\ \frac{KT\,(c+Jm\,s)}{KT\,KV + (c+Jm\,s)(R+Rm+La\,s)} & \frac{KT\,KV}{KT\,KV + (c+Jm\,s)(R+Rm+La\,s)} \end{pmatrix}$$

If we are interested In the transfer function between $\dot{\theta} = y_1$ and $V_a = u_1$, we would then choose the (1,1) element of the transfer function matrix.

```
GV[s_] = G[s][[1, 1]];

Print["G_V(s)"]
GV[s]
Print[""]

Print["Expanded Denominator"]
Collect[Expand[Denominator[GV[s]]], {s, s^2}]
```

$G_V(s)$

$$\frac{KT}{KT\,KV + (c + Jm\,s)(R + Rm + La\,s)}$$

Expanded Denominator

$$KT\,KV + c\,R + c\,Rm + (c\,La + Jm\,R + Jm\,Rm)\,s + Jm\,La\,s^2$$

Matlab provides a function, 'ss2tf' to do this conversion for you.

So we have

$$G_V(s) = \frac{\dot{\theta}(s)}{V_a(s)} = \frac{K_T}{J_m L_a s^2 + (c L_a + J_m R + J_m R_m) s + (K_T K_V + c R + c R_m)}$$

Using approximate numerical values we can write this as (see Matlab script)

$$G_V(s) = \frac{0.09854}{2.135 * 10^{-6} s^2 + 0.002179 s + 0.01034} = \frac{46\,163}{s^2 + 1021 s + 4845}$$

Note that some transfer functions may look to be different but are actually the same. You can examine the differences using the 'zpk' function in Matlab.

Cleaning Up Numerical Error in Transfer Functions

Note that in many situations, a transfer function object in Matlab may not perform pole/zero cancellation properly. To illustrate this, consider the following transfer function

$$P(s) = \frac{s^2 + 15.0001 s + 50.0005}{s^3 + 16 s^2 + 68 s + 80}$$

```
P[s_] = (s^2 + 15.0001 s + 50.0005)/(s^3 + 16 s^2 + 68 s + 80);
```

Note that the poles of and zeros of this transfer function can be found by finding roots of the denominator and numerator, respectively

```
Solve[Denominator[P[s]] == 0, s]
Solve[Numerator[P[s]] == 0, s]
```

$$\{\{s \to -10\}, \{s \to -4\}, \{s \to -2\}\}$$

$$\{\{s \to -10.0001\}, \{s \to -5.\}\}$$

The problem is that there is a pole at -10 and a zero at -10.0001. There are extremely close together but they are not the exact same so Matlab and Mathematica does not perform the pole/zero cancellation.

To allow pole/zero cancellation to occur if poles and zeros are within a range of each other, we developed the CleanUpTransferFunction.m function (see website).

Alternatively, you can use the Matlab function 'minreal', we will cover this in a later video/lecture.

Using Matlab

Go over different representations of a system

'tf'
'ss'
'tf2zpk'
'ss2tf'