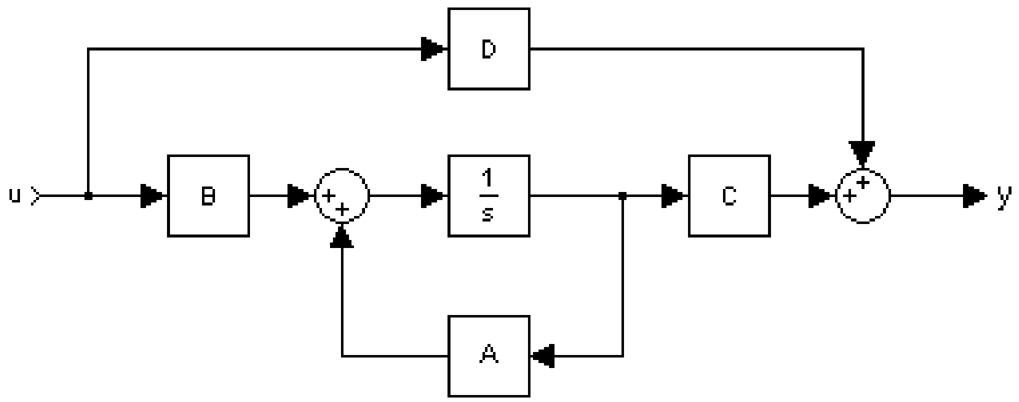
State Space Equations

There are several ways to describe a system of linear differential equations. The state-space representation is given by the equations:

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

x(.) is called the "state vector", y(.) is called the "output vector", u(.) is called the "input (or control) vector", A(.) is the "state matrix", B(.) is the "input matrix", C(.) is the "output matrix", and D(.) is the "feedthrough (or feedforward) matrix". For simplicity, D(.) is often chosen to be the zero matrix, i.e. the system is chosen not to have direct feedthrough. Notice that in this general formulation all matrixes are supposed time-variant, i.e. some or all their elements can depend on time.



A typical state space model

Controllability and observability

A continuous time-invariant state-space model is **controllable** if and only if

$$rank\begin{bmatrix} B & AB & \dots & A^{n-1}B \end{bmatrix} = n$$

A continuous time-invariant state-space model is **observable** if and only if

$$rank\begin{bmatrix} C \\ CA \\ \dots \\ CA^{n-1} \end{bmatrix} = n$$

(Rank is the number of linearly independent rows in a matrix.)

Any given transfer function can easily be transferred into state-space by the following approach:

Given a transfer function, expand it to reveal all coefficients in both the numerator and denominator. This should result in the following form:

$$\mathbf{G}(s) = \frac{n_1 s^3 + n_2 s^2 + n_3 s + n_4}{s^4 + d_1 s^3 + d_2 s^2 + d_3 s + d_4}.$$

The coefficients can now be inserted directly into the state-space model by the following approach:

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -d_1 & -d_2 & -d_3 & -d_4 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \mathbf{u}(t)$$

$$\mathbf{y}(t) = \begin{bmatrix} n_1 & n_2 & n_3 & n_4 \end{bmatrix} \mathbf{x}(t).$$

This state-space realization is called **controllable canonical form** because the resulting model is guaranteed to be controllable.

The transfer function coefficients can also be used to construct another type of canonical form

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -d_1 & 1 & 0 & 0 \\ -d_2 & 0 & 1 & 0 \\ -d_3 & 0 & 0 & 1 \\ -d_4 & 0 & 0 & 0 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ n_4 \end{bmatrix} \mathbf{u}(t)$$

$$\mathbf{y}(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \mathbf{x}(t).$$

This state-space realization is called **observable canonical form** because the resulting model is guaranteed to be observable.

Enter the system matrices :

```
A = [   0   1      0
     980   0   -2.8
       0   0   -100];
B = [0
     0
     100];
C = [1    0    0];
```

One of the first things you want to do with the state equations is find the poles of the system; these are the values of s where det(sI - A) = 0, or the eigenvalues of the A matrix:

```
poles = eig(A)
```

You should get the following three poles:

```
poles =

  31.3050
 -31.3050
-100.0000
```

One of the poles is in the right-half plane, which means that the system is unstable in open-loop.

In order to enter a state space model into MATLAB, the variables much be given numerical value, because MATLAB cannot manipulate symbolic variables without the symbolic toolbox. Enter the coefficient matrices **A**, **B**, **C**, and **D** into MATLAB. The syntax for defining a state space model in MATLAB is:

statespace = ss(A, B, C, D)

where **A**, **B**, **C**, and **D** are from the standard vector-matrix form of a state space model.

EXAMPLE

Our state vector consists of two variables, *x* and *v* so our vector-matrix will be in the form:

$$\begin{bmatrix} x' \\ v' \end{bmatrix} = \mathbf{A}\begin{bmatrix} x \\ v \end{bmatrix} + \mathbf{B}f(t)$$

$$y = \mathbf{C}\begin{bmatrix} x \\ v \end{bmatrix} + \mathbf{D}f(t)$$

The first row of **A** and the first row of **B** are the coefficients of the first state equation for x'. Likewise the second row of **A** and the second row of **B** are the coefficients of the second state equation for v'. **C** and **D** are the coefficients of the output equation for y. This yields:

$$\begin{bmatrix} x' \\ v' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} f(t)$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} f(t)$$

Therefore,

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 1 \\ 1/m \end{bmatrix}$$

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

$$\mathbf{D} = \begin{bmatrix} 0 \end{bmatrix}$$

Input the tutorial state space model into MATLAB:

Theoretical values must be determined for the constants m, b, and k. For the sake of example, we will take m = 2, b = 5, and k = 3.

```
>> m = 2;

>> b = 5;

>> k = 3;

>> A = [ 0  1 ; -k/m -b/m ];

>> B = [ 0 ; 1/m ];

>> C = [ 1 0 ];

>> D = 0;

>> my_ss = ss(A, B, C, D)
```

This assigns the state space model under the name tutorial_ss and output the following:

a =

	x1	*x2*
x1	*0*	*1*
x2	*-1.5*	*-2.5*

b =

	u1
x1	*0*
x2	*0.5*

c =

	x1	*x2*
y1	*1*	*0*

d =

	u1
y1	*0*

Continuous-time model.

In order to extract the A, B, C, and D matrices from a previously defined state space model, use MATLAB's ssdata command.

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

where statespace is the name of the state space system.

Extract the A, B, C, and D matrices from tutorial_ss using MATLAB:

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

The MATLAB output will be:

A =

-2.5000 -0.3750
4.0000 0

B =

0.2500
0

C =

0 0.5000

D =

0

Remember how we construct a transfer dunction using Matlab :

my_tf = tf (num, den)

EXAMPLE

Input the transfer function **X**(s)/**F**(s) = 1/[$ms^2 + bs + k$] *into MATLAB:*

For illustration purposes, this example uses $m = 2$, $b = 5$, and $k = 3$.

```
>> m = 2;
>> b = 5;
>> k = 3;
>> num = [ 1 ];
>> den = [ m b k ];
>> my_tf = tf(num, den)
```

MATLAB will assign the transfer function under the name tutorial_tf, and output the following:

Transfer function:

```
      1
---------------
2 s^2 + 5 s + 3
```

Using this representation, MATLAB can determine the transfer function from the state space representation in two ways. To find the numerator matrix and denominator vector of the transfer function of the system from the matrices of the state space model use MATLAB's ss2tf command:

[num, den] = ss2tf(A, B, C, D)

where num is defined as the numerator matrix, and den is defined as the denominator matrix, and A, B, C, and D are the coefficients of the state space model.

In order to find the entire transfer function system from the state space model, use the following command:

transferfunction = tf(statespace)

where statespace is the name of the state space system, and transferfunction is the name of the generated transfer function system.

Find A, B, C, and D, the state space vectors of my_tf *using MATLAB:*

```
>> [A, B, C, D] = tf2ss(num,den)
```

The MATLAB output will be:

A =

-2.5000 -1.5000
1.0000 0

B =

1
0

C =

0 0.5000

D =

0

Find the state space system of tutorial_tf *using MATLAB:*

```
>> my_ss = ss(my_tf)
```

MATLAB will assign the state space system der the name tutorial_ss, and output the following:

```
a =
        x1  x2
 x1  -2.5 -0.375
 x2  4       0

b =
        u1
 x1  0.25
 x2  0

c =
       x1 x2
  y1  0   0.5

d =
       u1
  y1   0

Continuous-time model.
```