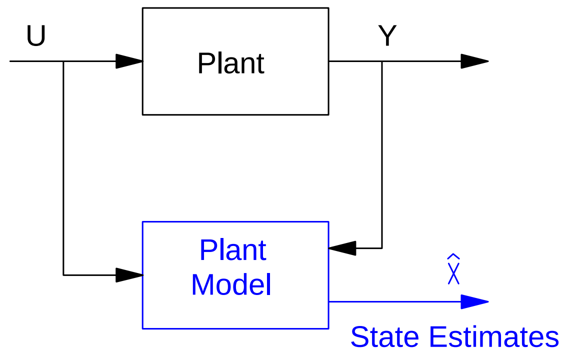
Full-Order Observers

Problem: The previous design requires measurements of *all* of the system's states. Sometimes you only have access to a few states. Can you estimate the states based upon the inputs, outputs, and dynamics of the system? If so, you could use these estimates of the states for the full-state feedback controllers we looked at previously.



Observer: Estimate the states based upon the input, output, and model of the plant

Solution: Given a system with states X:

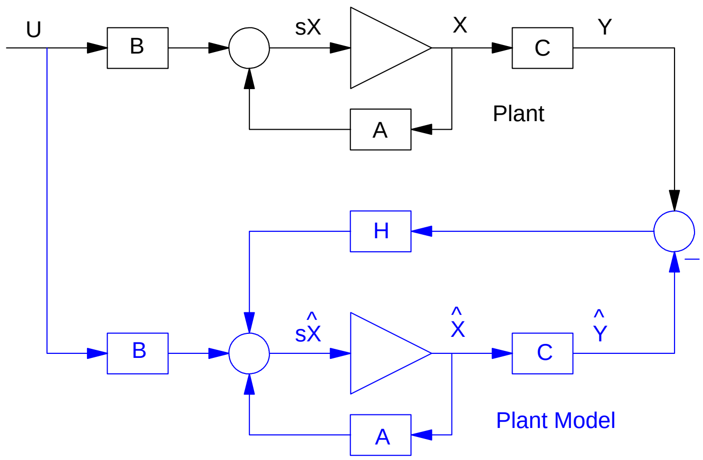
$$sX = AX + BU$$

$$Y = CX$$

Define a model of the system with estimates of the states, $\hat{X}$. Add a term based upon the error between the plant's output (Y) and it's estimate.

$$s\hat{X} = A\hat{X} + BU + H\left(Y - \hat{Y}\right)$$

$$\hat{Y} = C\hat{X}$$



Plant along with it's model (observer)

Define the error between the states and their estimate:

$$E = X - \hat{X}$$

If you can drive the error to zero, you know that the estimates are the same as the states.

Take the difference between the dynamics

$$sX - s\hat{X} = (AX + BU) - \left(A\hat{X} + BU\right) - H\left(Y - \hat{Y}\right)$$

Do some algebra to find the dynamics of the error:

$$s\left(X - \hat{X}\right) = A\left(X - \hat{X}\right) + BU - BU - H\left(CX - C\hat{X}\right)$$

$$sE = AE - HCE$$

$$\boxed{sE = (A - HC)E}$$

Note what this says:

- Presumably, there is an initial condition on E. That just means your initial guess on the states will be wrong.
- If you can make (A - HC) stable, the error will be driven to zero.
- Furthermore, the error between the actual states and state estimates has no input - meaning this system is uncontrollable. That is actually good: the error goes to zero no matter what you do with the input.

Problem: How to you select the gain matrix, H?

Solution: Essentially, the problem is how to select the gain, H, so that the poles of (A-HC) at some desired spot:

$$eig(A - HC) = \{\lambda_1, \lambda_2, \lambda_3, \cdots\}$$

Note that the eigenvalues of A are identical to the eigenvalues of A^T. Taking the transpose

$$eig(A - HC)^T = \{\lambda_1, \lambda_2, \lambda_3, \cdots\}$$

$$eig(A^T - C^T H^T) = \{\lambda_1, \lambda_2, \lambda_3, \cdots\}$$

This is identical to the previous problem of placing the poles of system (A, B)

$$eig(A - BK_x) = \{\lambda_1, \lambda_2, \lambda_3, \cdots\}$$

Likewise, you can use Bass-Gura to place the poles of (A^T, C^T). The result (Kx) will actually be HT.

Note that in order to use Bass-Gura, the controllability matrix must be full rank (i.e. the system must be controllable)

$$\rho[B, AB, A^2B, \ldots, A^{N-1}B] = N$$

Substituting (A^T, C^T) for (A, B) results in

$$\rho\left[C^T, A^TC^T, \ldots (A^T)^{N-1}C^T\right] = N$$

Since the rank of a matrix is equal to the rank of the transpose of the matrix,

$$\boxed{\rho\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{N-1} \end{bmatrix} = N}$$

This is termed the *observability* matrix. The requirement for being able to estimate the system's states is that the system is *observable.*

Once you're done, the system becomes

$$sX = AX + BU$$

$$s\hat{X} = A\hat{X} + BU + H\left(Y - \hat{Y}\right)$$

Combining, the augmented system becomes:

$$s\begin{bmatrix} X \\ \hat{X} \end{bmatrix} = \begin{bmatrix} A & 0 \\ HC & A - HC \end{bmatrix}\begin{bmatrix} X \\ \hat{X} \end{bmatrix} + \begin{bmatrix} B \\ B \end{bmatrix}U$$

$$Y = \begin{bmatrix} C & 0 \end{bmatrix}\begin{bmatrix} X \\ \hat{X} \end{bmatrix}$$

Example 1:

Assume you can only measure the 4th state for a heat equation. Design a full-order observer to estimate all four states

$$sX = \begin{bmatrix} -2 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}X + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}U$$

$$Y = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}X$$

Solution: First, check that the system is observable (i.e. that it can be done)

```
-->rank([C; C*A; C*A*A; C*A*A*A])

    4.
```

Yup - the observability matrix is full rank. You can place the observer poles wherever you like.

Place the observer poles at {-1, -2, -3, -4}

```
-->H = ppl(A', C', [-1, -2, -3, -4])'

    1.
    2.
    2.
    3.

-->eigc(A-H*C)

  - 1.
  - 2.
  - 4.
  - 3.
```

The augmented system (plant plus observer) is then

```
-->A8 = [A, zeros(4,4); H*C, A-H*C]

  - 2.    1.    0.    0. :   0.    0.    0.    0.
    1.  - 2.    1.    0. :   0.    0.    0.    0.
    0.    1.  - 2.    1. :   0.    0.    0.    0.
    0.    0.    1.  - 1. :   0.    0.    0.    0.
   - - - - - - - - - - - - - - - - - - - - - - - -
    0.    0.    0.    1. : - 2.    1.    0.  - 1.
    0.    0.    0.    2. :   1.  - 2.    1.  - 2.
    0.    0.    0.    2. :   0.    1.  - 2.  - 1.
    0.    0.    0.    3. :   0.    0.    1.  - 4.

-->B8 = [B; B]
 B8  =

    1.
    0.
    0.
    0.
  - - -
    1.
    0.
    0.
    0.
```

The poles of the augmented system are the poles of the plant (A) and the observer poles:

```
-->eig(A8)
```

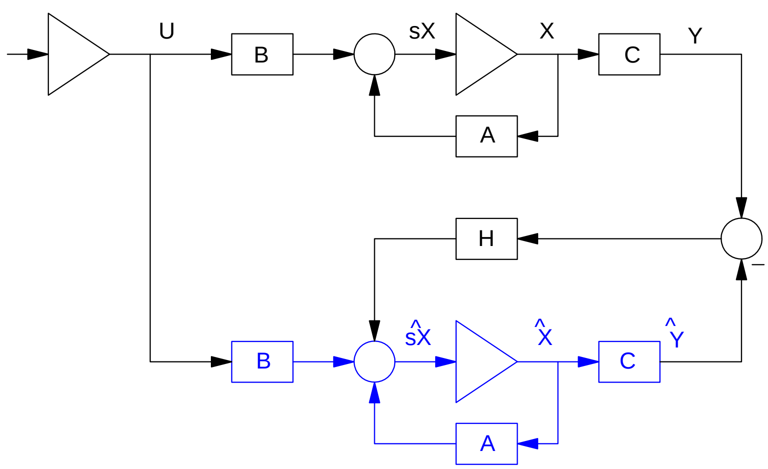
```
- 0.1206148
- 4.
- 3.5320889
- 3.
- 2.3472964
- 2.
- 1.
- 1.
```

Note that the augmented system is uncontrollable:

```
-->rank([B8,A8*B8,(A8^2)*B8,(A8^3)*B8,(A8^4)*B8,(A8^5)*B8,(A8^6)*B8,(A8^7)*B8])

    4.
```

That's actually expected: the error in the observer states goes to zero regardless of what the input, U, does. The error in the observer estimates is uncontrollable. (that's good).

To simulate, add a state to create the step input (U)



System for simulating the plant and observer.
An additional integrator is used for U to allow initial conditions along with a step input.

Simulate the following 8x8 system with initial conditions:

$$s\begin{bmatrix} X \\ \hat{X} \end{bmatrix} = \begin{bmatrix} A & 0 \\ HC & A-HC \end{bmatrix}\begin{bmatrix} X \\ \hat{X} \end{bmatrix}$$

Give it an initial condition (X = 0, estimate = [0.2, 0.4, 0.6, 0.8] so you can see the error driven to zero)

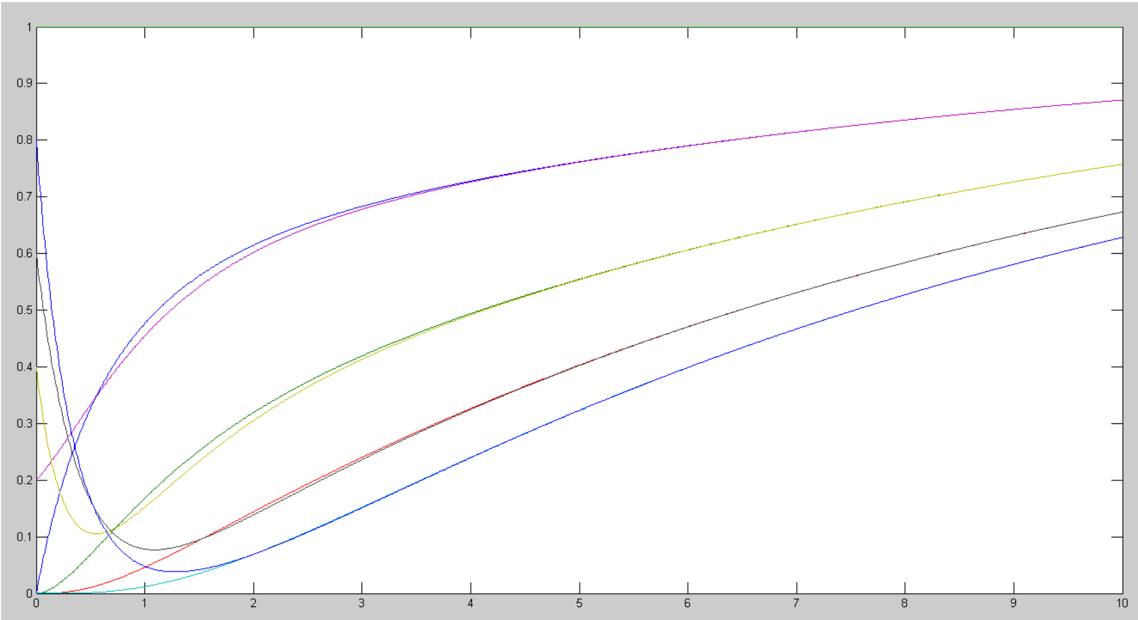
```
>> X0 = [0;0;0;0;  0.2;0.4;0.6;0.8]

         0
         0     initial states on the plant
         0
         0
  - - - - -
    0.2000
    0.4000     initial states for the model
    0.6000
    0.8000
```

Output all eight states:

```
>> C8 = eye(8,8);
>> D8 = zeros(8,1);
```

Simulate using the function step2:

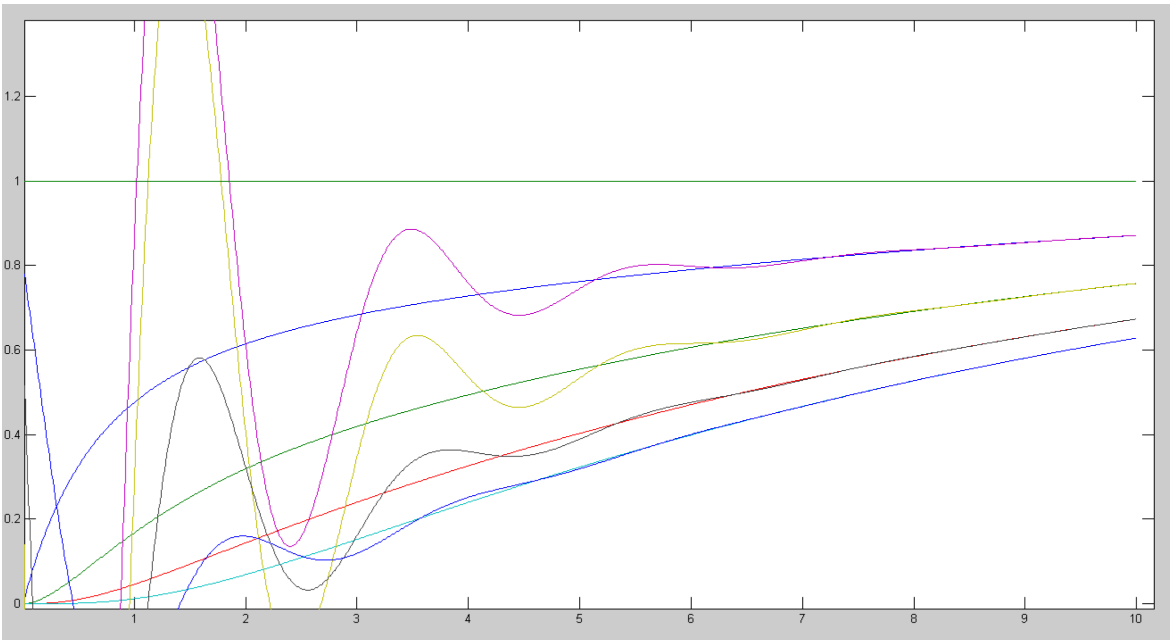
```
>> y = step2(A8, B8, C8, D8, X0, t);
>> plot(t,y);
```



Plant states and their estimates. Note that the estimates converge to the plant states in about 4 seconds.
eig(A - HC) = { -1, -2, -3, -4 }

You can even make the observer poles complex if you like:

```
-->H = ppl(A', C', [-1+j*3, -1-j*3, -3, -4])'

    28.
    28.
    9.
    2.

-->A8 = [A, zeros(4,4); H*C, A-H*C]
-->B8 = [B; B]
>> C8 = eye(8,8);
>> D8 = zeros(8,1);

>> X0 = [0;0;0;0;  0.2;0.4;0.6;0.8]
>> y = step2(A8, B8, C8, D8, x0, t);
>> plot(t,y);
```



Plant states and their estimates. Note that the estimates converge to the plant states in about 4 seconds.
eig(A - HC) = { -1 + j3, -1 - j3, -3, -4 }

Example 2: Gantry System

Design a full-order observer for an inverted pendulum:

$$s\begin{bmatrix} x \\ \theta \\ sx \\ s\theta \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 19.6 & 0 & 0 \\ 0 & -29.4 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ sx \\ s\theta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \end{bmatrix}F$$

Option 1: Use measurement of angle (θ)

$$Y = \theta = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ sx \\ s\theta \end{bmatrix}$$

Check that the system is observable from angle:

```
-->A = [0,0,1,0;0,0,0,1;0,19.6,0,0;0,-29.4,0,0]
-->C = [0,1,0,0]
-->rank([C; C*A; C*A*A; C*A*A*A])

    2.
```

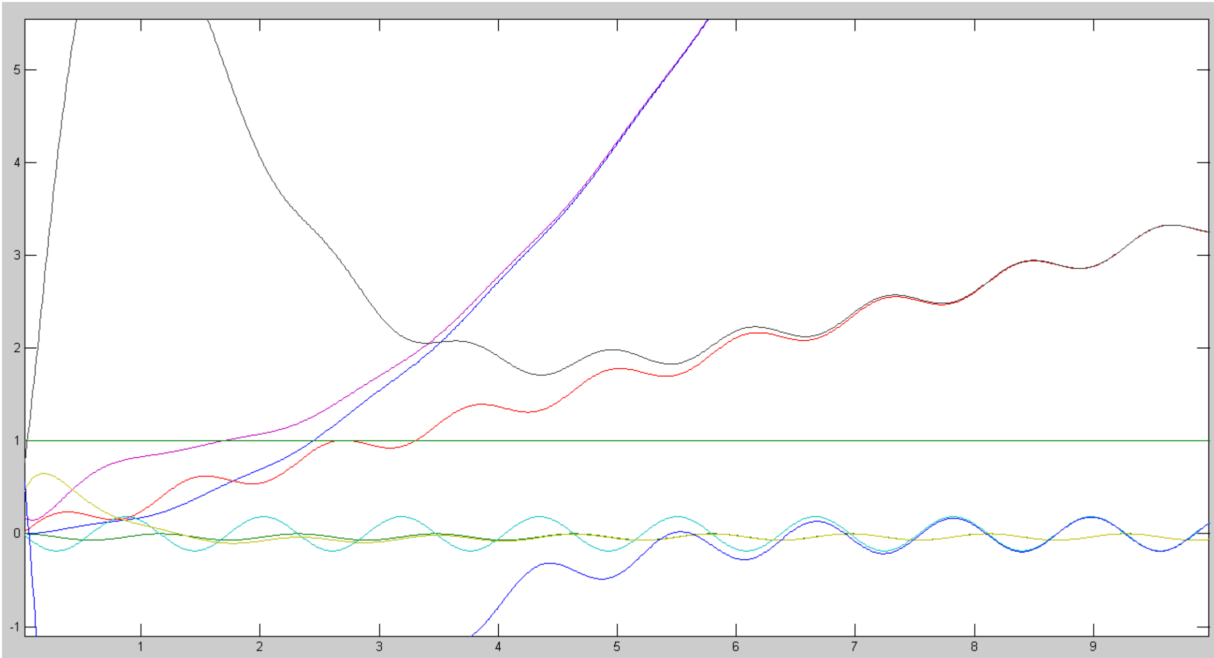
You cannot estimate all four states just by measuring only the angle.

Option 2: Use measurements of position (x)

$$Y = x = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ sx \\ s\theta \end{bmatrix}$$

```
-->A = [0,0,1,0;0,0,0,1;0,19.6,0,0;0,-29.4,0,0]
-->C = [1,0,0,0]

    1.    0.    0.    0.

-->rank([C; C*A; C*A*A; C*A*A*A])

    4.

-->H = ppl(A', C', [-1, -2, -3, -4])'

    10.
  - 12.44898
    5.6
  - 7.1755102
```

```
>> eig(A - H*C)

   -4.0000
   -3.0000
   -2.0000
   -1.0000

>>  A8 = [A, zeros(4,4) ;
         H*C, A-H*C ]

         plant                   model
    0     0     1     0 :    0     0     0     0
    0     0     0     1 :    0     0     0     0
    0  19.6     0     0 :    0     0     0     0
    0 -29.4     0     0 :    0     0     0     0
 - - - - - - - - - - - - - - - - - - - - - - - -
   10     0     0     0 :  -10     0     1     0
-12.4     0     0     0 : 12.4     0     0     1
  5.6     0     0     0 : -5.6  19.6     0     0
 -7.1     0     0     0 :  7.1 -29.4     0     0
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -


>> X0 = [0;0;0;0;  0.2;0.4;0.6;0.8]

         0
         0   initial condition for the plant
         0
         0
- - - - - - - -
    0.2000
    0.4000   initial condistion for the estimator
    0.6000
    0.8000


>> X0 = [0;0;0;0;  0.2;0.4;0.6;0.8]
>> y = step2(A8, B8, C8, D8, X0, t);
>> plot(t,y);
```



States and their estimates: Note that the estimates converge to the plant's states in about 7 seconds with no overshoot
eig(A - HC) = { -1, -2, -3, -4 }

function Step2.m

```
function [ y ] = step2( A, B, C, D, X0, t)

npt = length(t);
[m,n] = size(C);

y = zeros(npt, m);

X = X0;

T = t(2) - t(1);

Ad = expm(A*T);
Bd = T*B;

y(1,:) = (C*X + D)';

for i=2:npt
    X = Ad*X + Bd;
    Y = C*X + D;
    y(i,:) = Y';
    end

end
```