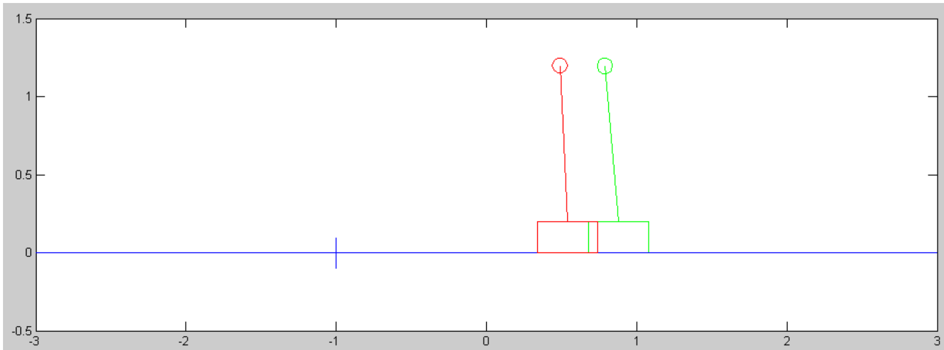
ECE 463: Homework #8

Linear Observers. Due Monday March 18th
Please submit as a hard copy, emai lto jacob.glower@ndsu.edu, or submit on BlackBoard



Cart and Pendulum from homework #4 with a state estimator (green)

Use the dynamics for the cart and pendulum from homework set #4

$$s\begin{bmatrix} x \\ \theta \\ \dot{x} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -2.45 & 0 & 0 \\ 0 & 9.42 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ \dot{x} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0.25 \\ -0.1923 \end{bmatrix} F$$

1) Design a full-state feedback control law of the form

$$U = F = K_r R - K_x X$$

so that the closed-loop system has

- A 2% settling time of 8 seconds, and
- 5% overshoot for a step input.

Plot the step response of the linarized system in Matlab.

Tranlation: Place the closed-loop poles at s = -0.5 + j0.5243. Doing so using pole placement:

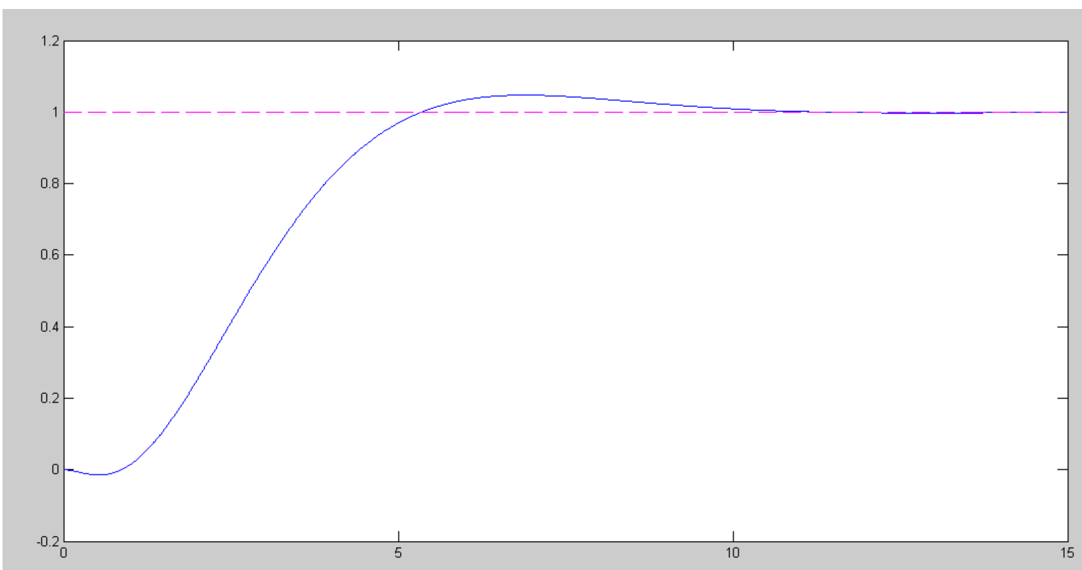
```
>> A = [0,0,1,0;0,0,0,1;0,-2.45,0,0;0,9.42,0,0]

         0         0    1.0000         0
         0         0         0    1.0000
         0   -2.4500         0         0
         0    9.4200         0         0

>> B = [0;0;0.25;-0.1923]

         0
         0
    0.2500
   -0.1923
```

```
>> C = [1,0,0,0];
>> Kx = ppl(A, B, [-0.5+j*0.5243,-0.5-j*0.5243,-2,-3])

Kx =   -1.6717 -111.0911   -4.5781  -37.1530

>> DC = -C*inv(A-B*Kx)*B

DC =   -0.5982

>> Kr = 1/DC

Kr =

   -1.6717
```

Plotting the step response of the closed-loop system:

```
>> G = ss(A - B*Kx, B*Kr, C, 0);
>> t = [0:0.01:15]';
>> y = step(G,t);
>> plot(t,y,'b',t,0*y+1,'m--')
>>
```

>>



Assume you can only measure the cart position and beam angle.

2) Design a full-order observer to estimate all four states so that the observer is 2-5 times faster than the plant. You may use either cart position or beam angle (or both) as measurements.

There is a problem here: our pole-placement algorithm can't handle two inputs (position and angle).

So, let's just use position...

```
>> C = [1,0,0,0];
>> H = ppl(A', C', [-1+j, -1-j, -3, -4])'

     9.0000
   -50.1143
    37.4200
  -153.6720
```

4) Give the state-space model of the closde loop system using the states:

$$U = F = K_r R - K_x X$$

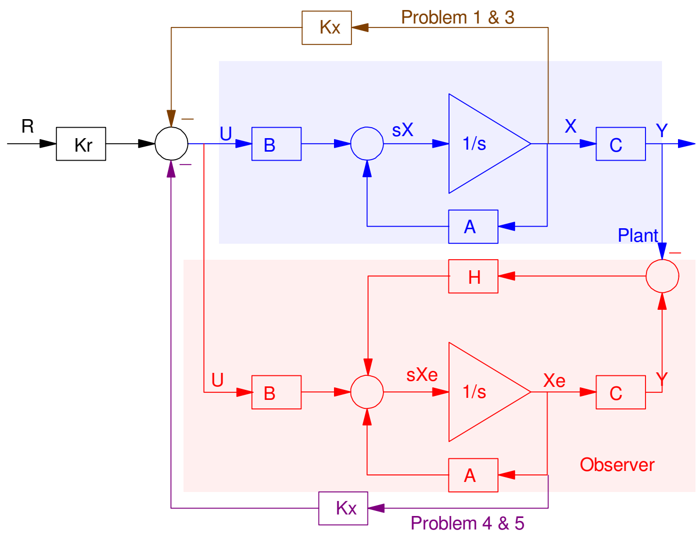
and plot the step response with initial conditions of

X(0) = [0, 0, 0, 0]' $X_{observer}(0)$ = [0.1, 0.1, 0.1, 0.1]'

(note: use the function step3)

The net system (plant + observer) is

$$\begin{bmatrix} sX \\ sX_e \end{bmatrix} = \begin{bmatrix} A - BK_x & 0 \\ HC - BK_x & A - HC \end{bmatrix} \begin{bmatrix} X \\ X_e \end{bmatrix} + \begin{bmatrix} BK_r \\ BK_r \end{bmatrix} R$$

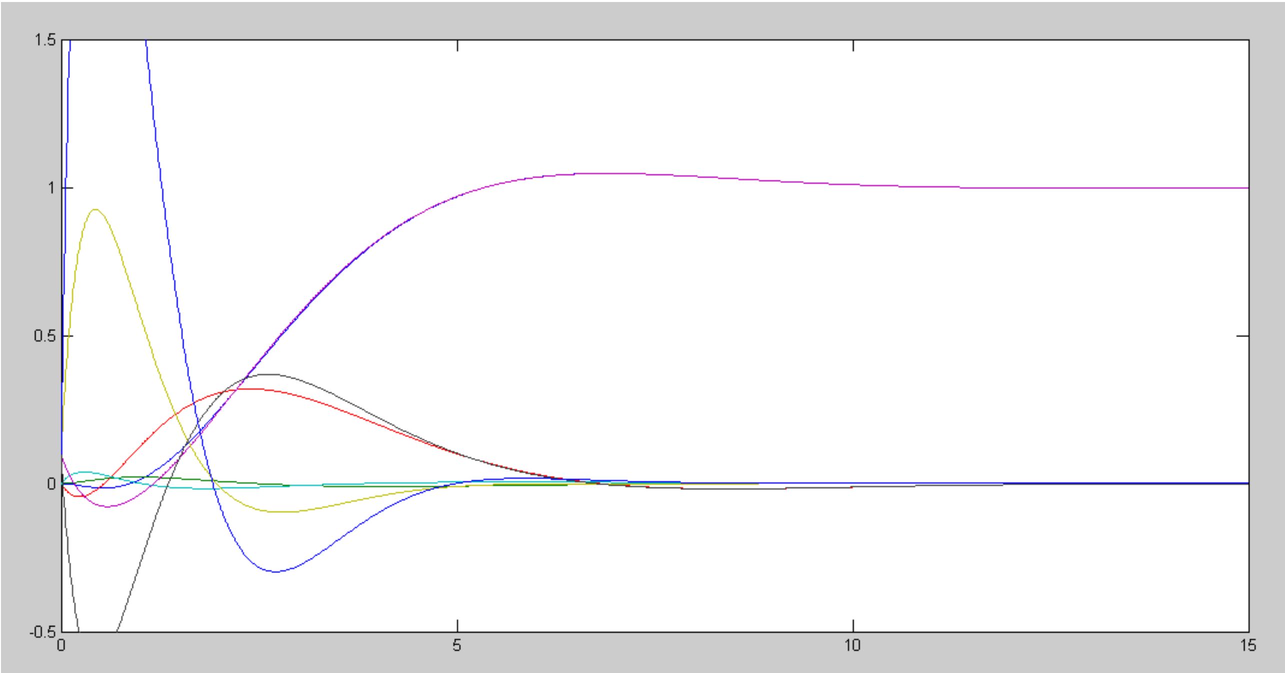


In Matlab

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

          0          0     1.0000          0          0          0          0          0
          0          0          0     1.0000          0          0          0          0
     0.4179    25.3228     1.1445     9.2882          0          0          0          0
    -0.3215   -11.9428    -0.8804    -7.1445          0          0          0          0
     9.0000          0          0          0    -9.0000          0     1.0000          0
   -50.1143          0          0          0    50.1143          0          0     1.0000
    37.8379    27.7728     1.1445     9.2882   -37.4200    -2.4500          0          0
  -153.9935   -21.3628    -0.8804    -7.1445   153.6720     9.4200          0          0

>> eig(A8)

  -4.0000
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
  -0.5000 + 0.5243i
  -0.5000 - 0.5243i
  -2.0000
  -3.0000
  -3.0000
```

Looks good - plant & observer & control law is stable

```
>> B8 = [B*Kr ; B*Kr];
>> C8 = eye(8,8);
>> D8 = zeros(8,1);
>> X0 = [0;0;0;0;0.1;0.1;0.1;0.1];
>> t = [0:0.01:15]';
>> R = 0*t + 1;
>> y = step3(A8, B8, C8, D8, t, X0, R);
>> plot(t,y)
>> ylim([-0.5,1.5])
>>
```



Step Response when Feeding Back Actual States

Note

- The plant behaves the same as problem #1: it should since we're feeding back the observer states
- The observer is kind of squirrelry for the first 5 seconds as it tries to figure out what the states are

4) Give the state-space model of the closde loop system using the state estimates:

$$U = F = K_r R - K_x X_e$$

and plot the step response with initial conditions of

$X(0) = [0, 0, 0, 0]'$ $X_{observer}(0) = [0.1, 0.1, 0.1, 0.1]'$

(note: use the function step3)

The net system (plant + observer) is

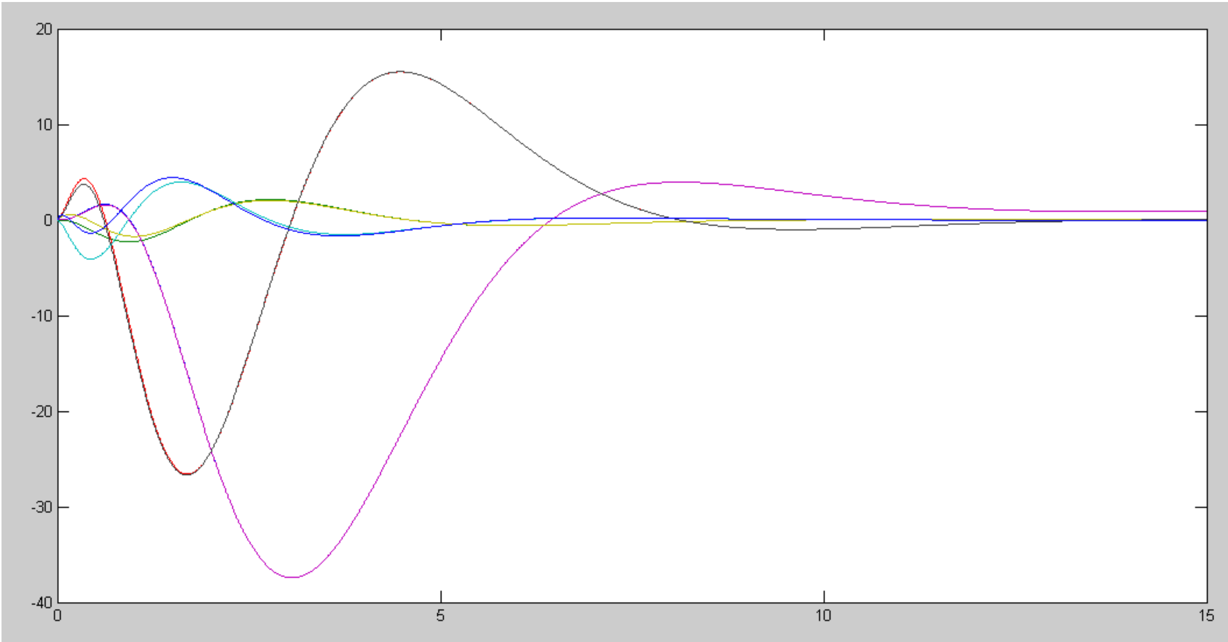
$$\begin{bmatrix} sX \\ sX_e \end{bmatrix} = \begin{bmatrix} A & -BK_x \\ HC & A - HC - BK_x \end{bmatrix} \begin{bmatrix} X \\ X_e \end{bmatrix} + \begin{bmatrix} BK_r \\ BK_r \end{bmatrix} R$$

Plotting the step response in Matlab:

```
>> A8 = [A, -B*Kx ; H*C, A-H*C-B*Kx]

A8 =

         0         0    1.0000         0         0         0         0         0
         0         0         0    1.0000         0         0         0         0
         0   -2.4500         0         0    0.4179   27.7728    1.1445    9.2882
         0    9.4200         0         0   -0.3215  -21.3628   -0.8804   -7.1445
    9.0000         0         0         0   -9.0000         0    1.0000         0
  -50.1143         0         0         0   50.1143         0         0    1.0000
   37.4200         0         0         0  -37.0021   25.3228    1.1445    9.2882
 -153.6720         0         0         0  153.3505  -11.9428   -0.8804   -7.1445

>> eig(A8)

  -0.5000 + 0.5243i
  -0.5000 - 0.5243i
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
  -4.0000
  -2.0000
  -3.0000 + 0.0000i
  -3.0000 - 0.0000i
```

The closed-loop poles are correct (observer & plant & feedback pole locations)

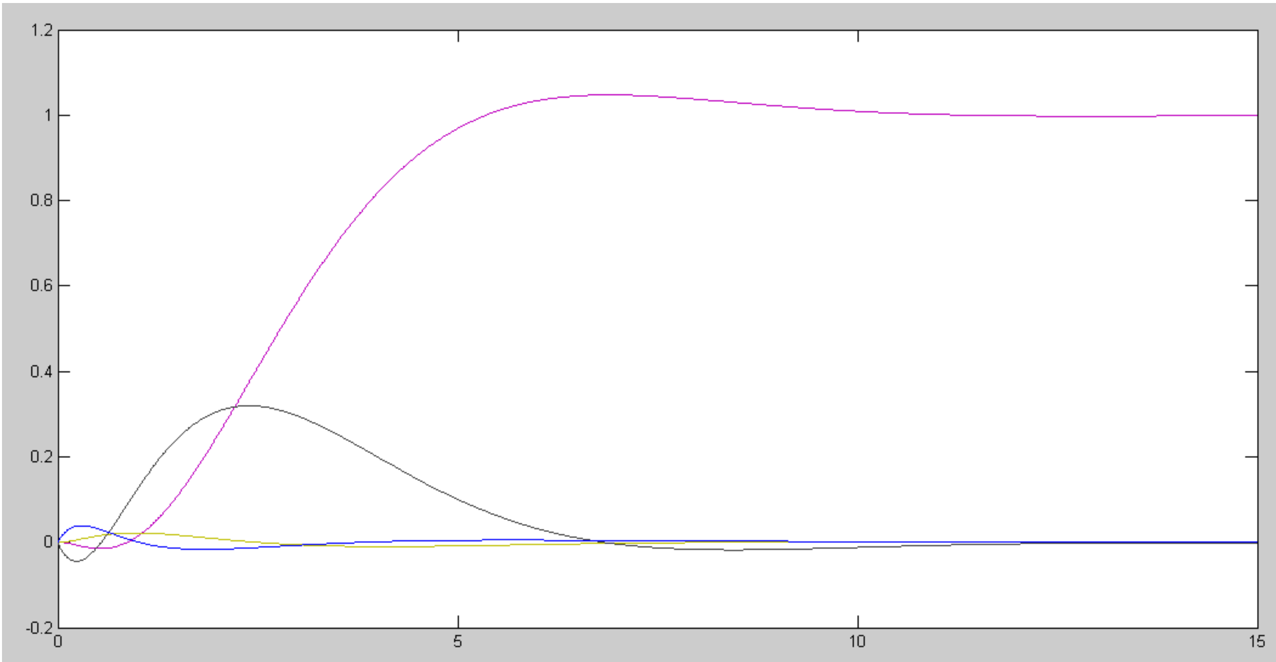
```
>> B8 = [B*Kr ; B*Kr]
>> C8 = eye(8,8);
>> D8 = zeros(8,1);
>> t = [0:0.01:15]';
>> X0 = [0;0;0;0;0.1;0.1;0.1;0.1];
>> R = 0*t+1;
>> y = step3(A8, B8, C8, D8, t, X0, R);
>> plot(t,y)
>>
```



The step response is pretty squirrely - due to the observer having bad estimates for the first four seconds.

If you start out with the observer states matching the plant states, it looks better:

```
>> X0 = zeros(8,1);
>> y = step3(A8, B8, C8, D8, t, X0, R);
>> plot(t,y)
>>
```

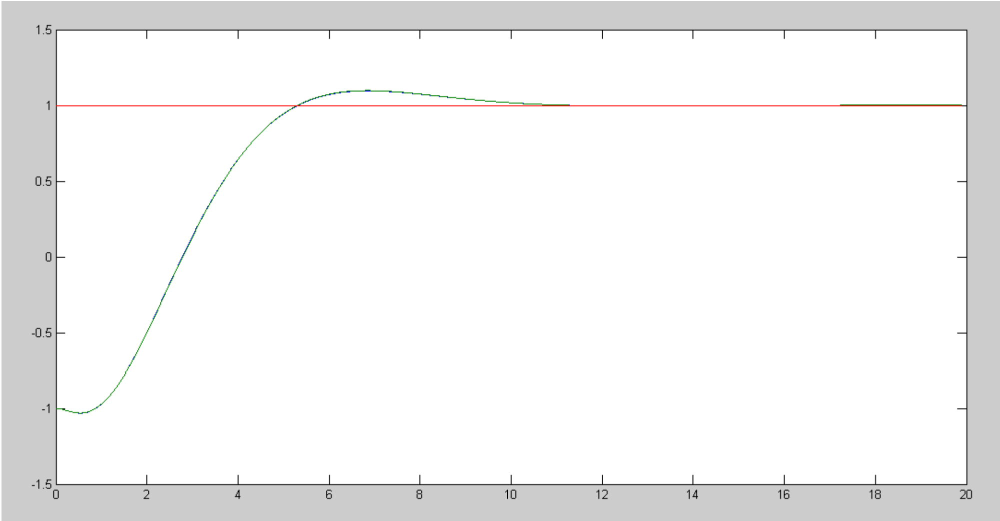


5) (20pt) Modify the cart and pendulum system to include

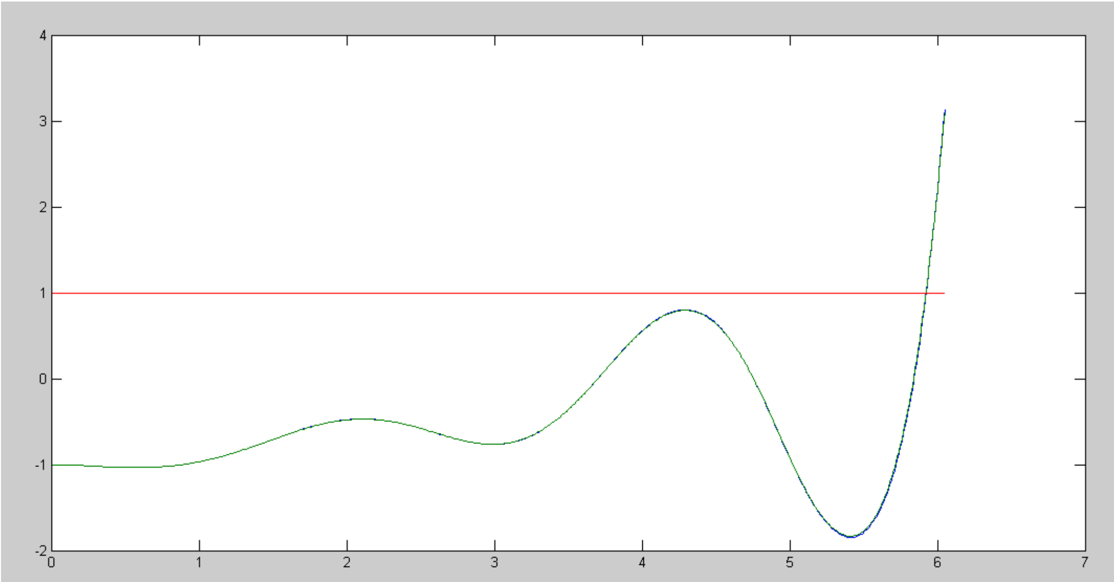
- your control law, and
- A full-order observer

Plot the step response of the nonlinear system + observer when

- Xe = $[0, 0, 0, 0]^T$
- Xe = $[0.1, 0.1, 0.1, 0.1]^T$



Step Response when feeding back actual states (plant & observer output)



Response when feeding back the state estimates

Sidelight: Just using position didn't work too well. Using trial and error, changing the C matrix to

```
C = [1, 10, 0, 0]
```

meaning I'm calling the output

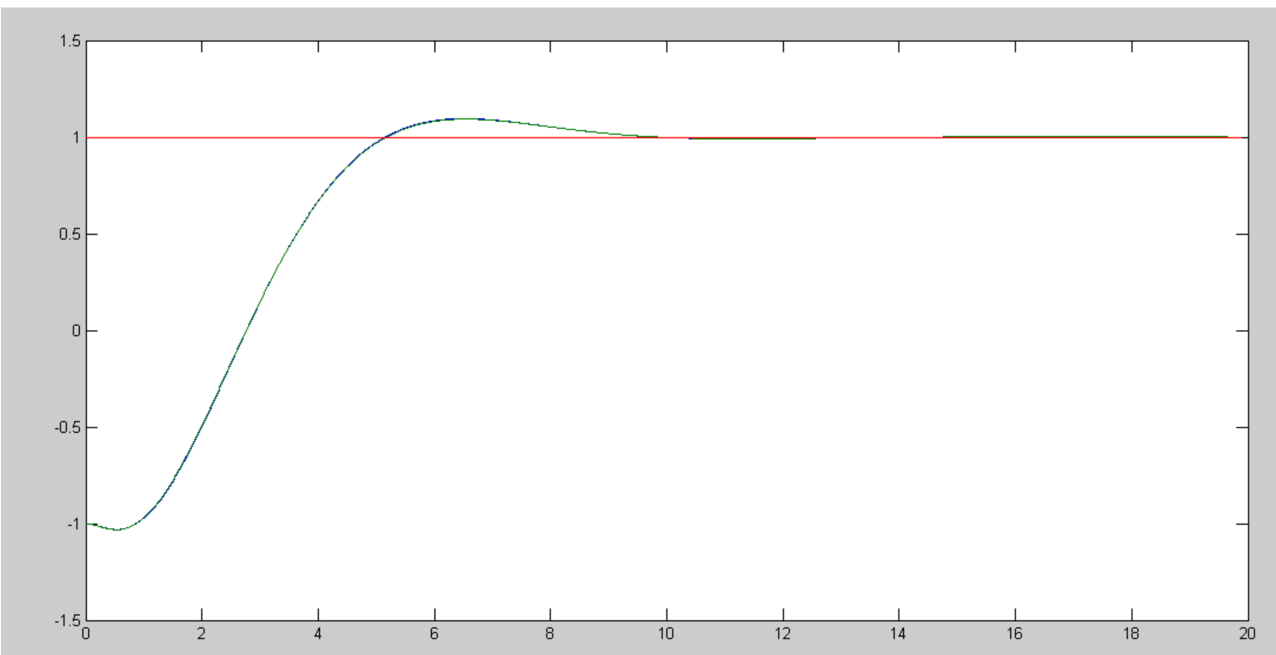
$$y = x + 10\theta$$

along with speeding up the observer poles to [-2, -3, -4, -5]

```
>> C = [1,10,0,0];
>> H = ppl(A', C', [-2, -3, -4, -5])'

  -17.1586
    3.1159
  -15.2265
    9.5646
```

results in a much better step response:

- just using position and angle measurements



It works better if you take into account the angle measurement...

Code:

```
% Cart and Pendulum
% Lecture %20
% Separation Principle

X = [-1,0,0,0]';
Ref = 1;
dt = 0.01;
t = 0;

% Control Law
Kx = [-1.6717 -111.0911   -4.5781  -37.1530];
Kr =  -1.6717;

% Full-Order Observer
Ae = [0,0,1,0;0,0,0,1;0,-2.45,0,0;0,9.42,0,0];
Be = [0;0;0.255;-0.192];
Ce = [1,10,0,0];
H = ppl(Ae', Ce', [-2, -3, -4, -5])';
Xe = X;

n = 0;
y = [];
while((t < 19.9) & (abs(X(1)) < 3))
   Ref = sign(sin(2*pi/10));
   U = Kr*Ref - Kx*Xe;
   dX = CartDynamics(X, U);
   dXe = Ae*Xe + Be*U + H*(Ce*X - Ce*Xe);

   X = X + dX * dt;
   Xe = Xe + dXe * dt;

   t = t + dt;
   n = mod(n+1, 5);
   if(n == 0)
      CartDisplay(X, Xe, Ref);
   end
   y = [y ; X(1), Xe(1), Ref];
end

hold off;
t = [1:length(y)]' * dt;
plot(t,y);
```