ECE 463/663 - Homework #9

Calculus of Variations. LQG Control. Due Wednesday, April 3rd
Please submit as a hard copy, email to jacob.glower@ndsu.edu, or submit on BlackBoard

Soap Film

1) Calculate the shape of a soap film connecting two rings around the X axis:

- Y(0) = 7
- Y(4) = 10

From the lecture notes, a soap film minimizes the surface area. The corresponding funtional is

$$J = \int \left(y\sqrt{1+\dot{y}^2} \right) dx$$

which has the solution

$$y = a \cdot \cosh\left(\frac{x-b}{a}\right)$$

Plugging in the two endpoints to solve for a and b

$$7 = a \cdot \cosh\left(\frac{-b}{a}\right)$$

$$10 = a \cdot \cosh\left(\frac{4-b}{a}\right)$$

Solving in Matlab, first create a cost function

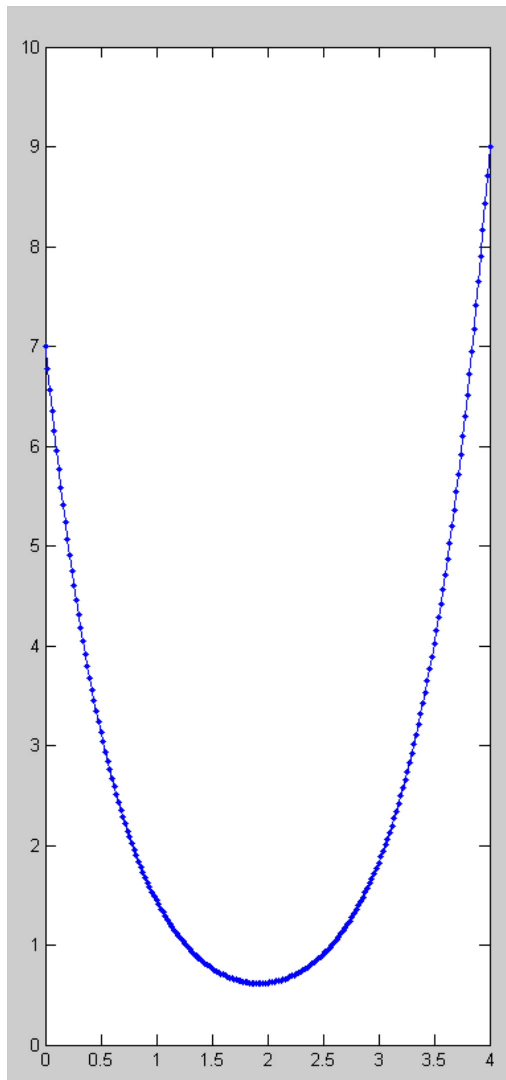
```
function [J] = soap(z)
   a = z(1);
   b = z(2);
   e1 = a*cosh(-b/a) - 7;
   e2 = a*cosh((4-b)/a) - 10;
   J = e1^2 + e2^2;
   end
```

Solve using fminsearch:

```
>> [z,e] = fminsearch('soap',[1,2])
          a          b
z =    0.6017    1.8924
e =  4.4168e-007
```

meaning

$$y = 0.6017 \cdot \cosh\left(\frac{x-1.8924}{0.6017}\right)$$



2) Calculate the shape of a soap film connecting two rings around the X axis:

- Y(0) = 7
- Y(2) = free

From the lecture notes,

$$y = a \cdot \cosh\left(\frac{x-b}{a}\right)$$

The endpoint constraint is

$$7 = a \cdot \cosh\left(\frac{-b}{a}\right)$$

The right endpoint constraint is

$$y' = -\sinh\left(\frac{2-b}{a}\right) = 0$$

Setting up a cost funiton in Matlab

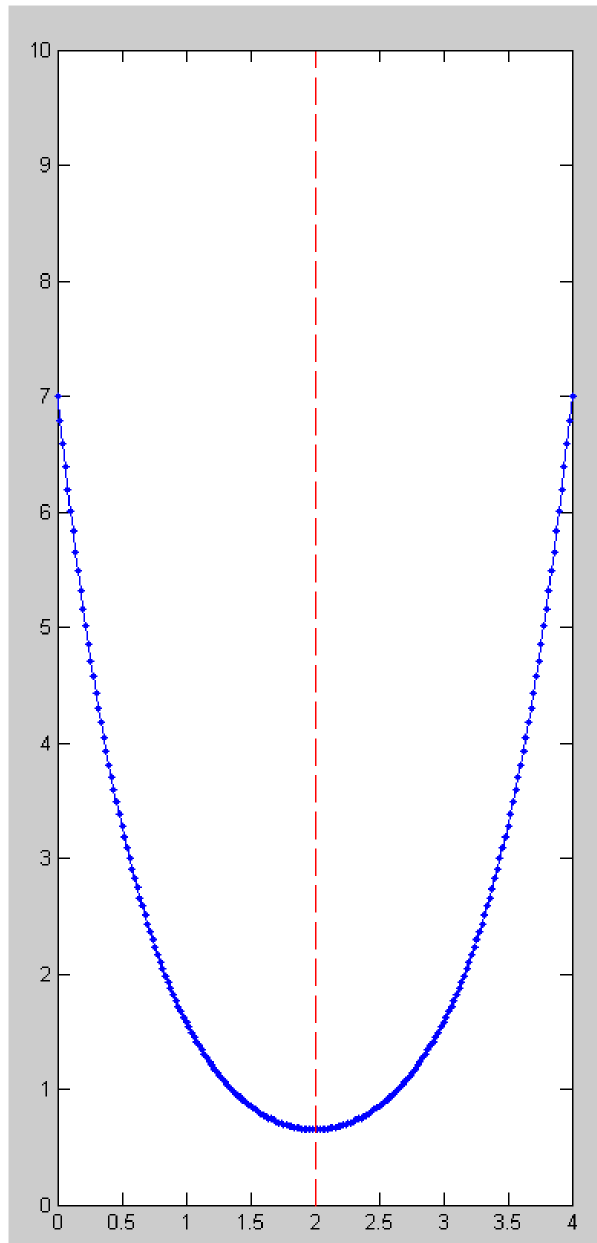
```
function [J] = soap(z)
   a = z(1);
   b = z(2);
   e1 = a*cosh(-b/a) - 7;
   e2 = sinh((2-b)/a);
   J = e1^2 + e2^2;
   end
```

Solving

```
>> [z,e] = fminsearch('soap',[1,2])

            a            b
z =    0.6529    2.0000
e =   6.7418e-009
```

so

```
>> a = z(1);
>> b = z(2);
>> y = a * cosh( (x-b)/a );
>> plot(x,y);
>> plot(x,y,[2,2],[0,10],'r--');
```



Hanging Chain

3) Calculate the shape of a hanging chain subject to the following constraints

- Length of chain = 13 meters
- Left Endpoint: (0,7)
- Right Endpoint: (10,5)

From the lecture notes, a hanging chain

- Minimizes the potential energy,
- With the constraint that the total lenngth is 12 meters

The corresponding funcitonal is

$$F = x\sqrt{1+\dot{y}^2} + M\sqrt{1+\dot{y}^2}$$

which results in the solution

$$y = a \cdot \cosh\left(\frac{x-b}{a}\right) - M$$

$$\left(a \cdot \sinh\left(\frac{x-b}{a}\right)\right)_0^{10} = 13$$

Set up a cost function

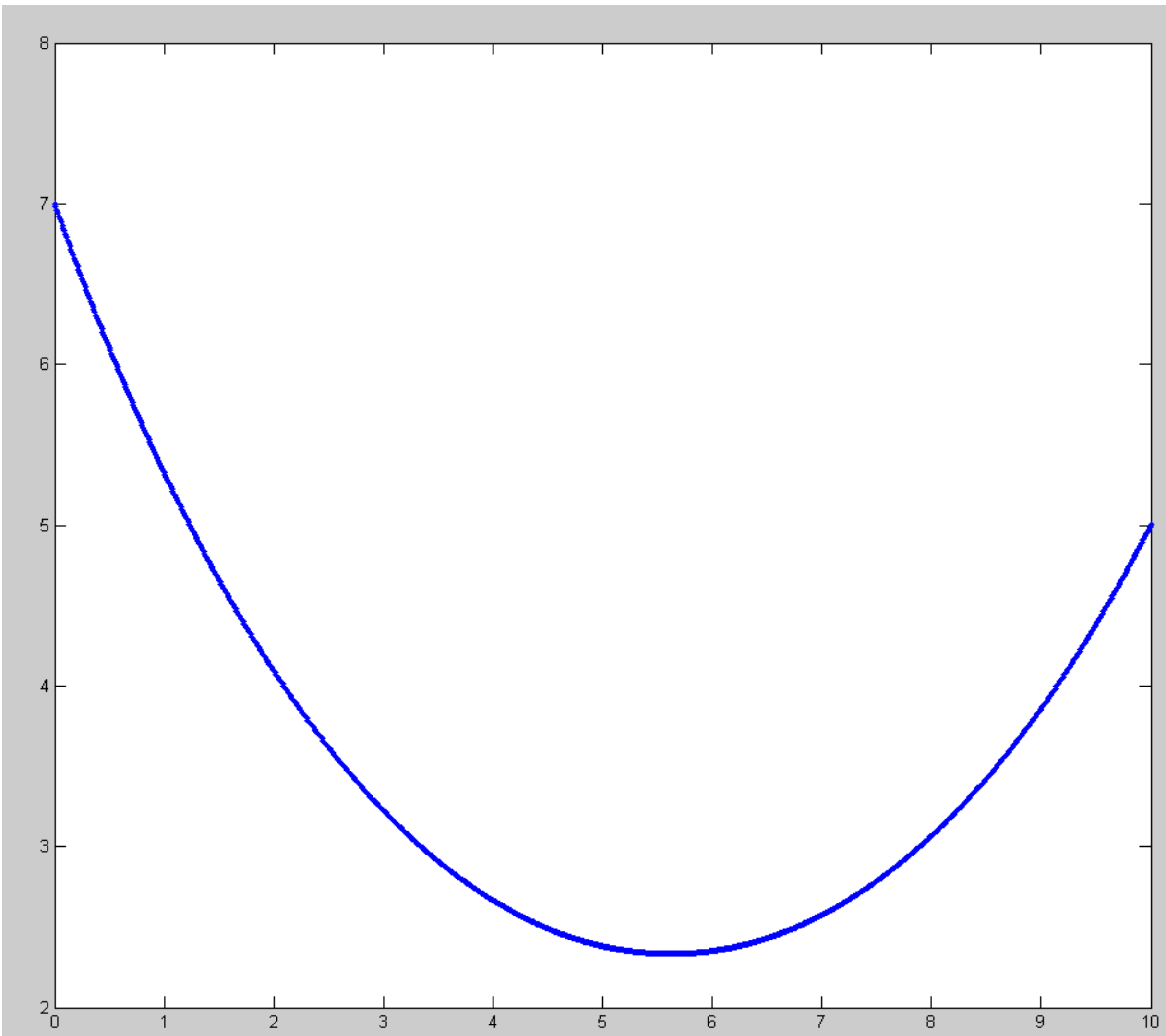
```
function J = chain(z)
a = z(1);
b = z(2);
M = z(3);

Length = 13;
x1 = 0;
y1 = 7;

x2 = 10;
y2 = 5;

e1 = a*cosh((x1-b)/a) - M - y1;
e2 = a*cosh((x2-b)/a) - M - y2;
e3 = a*sinh((x2-b)/a) - a*sinh((x1-b)/a) - Length;

J = e1^2 + e2^2 + e3^2;

end
```

Solving

```
>> [z,e] = fminsearch('chain',[1,2,3])

          a          b          M
z =     3.9805    5.6173    1.6471

e =  4.8029e-009
```

plotting the shape

```
>> a = z(1);
>> b = z(2);
>> M = z(3);
>> x = [0:0.01:10]';
>> y = a*cosh( (x-b)/a ) - M;
>> plot(x,y);
>> ylim([0,10])
```



Ricatti Equation

4) Find the function, x(t), which minimizes the following funcional

$$J = \int_0^{10} (2x^2 + 5\dot{x}^2) dt$$

x(0) = 6

x(10) = 7

Any funciton that minimizes this functional must minimize the Euler LaGrange equation

$$F_x - \frac{d}{dt}(F_{x'}) = 0$$

Solving

$$(4x) - \frac{d}{dt}(10\dot{x}) = 0$$

$$10\ddot{x} - 4x = 0$$

$$(10s^2 - 4)X = 0$$

Either

- x = 0, or
- s = {+0.6325, -0.0.6325}

going with the latter solution

$$x(t) = ae^{0.6325t} + be^{-0.6325t}$$

Plugging in the endpoint constraints

$$6 = a + b$$

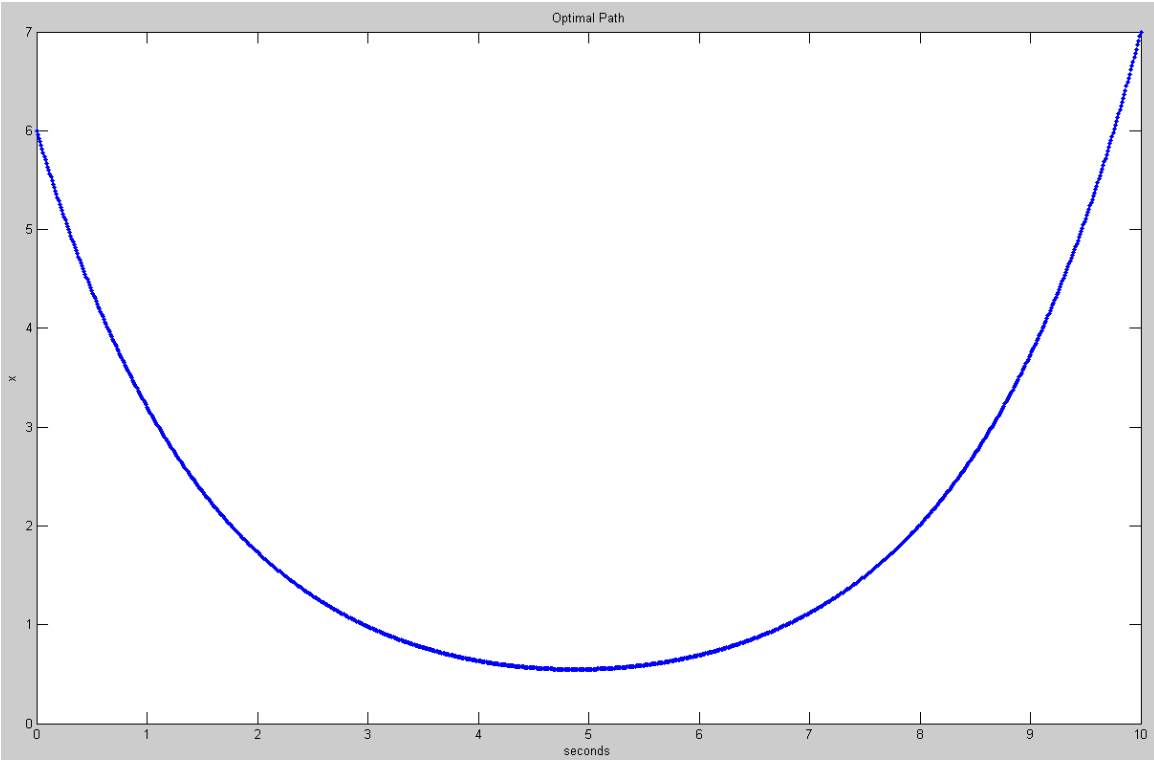
$$7 = 558.35a + 0.0018b$$

Solving 2 equations for 2 unknowns

```
>> A = [1,1 ; exp(5),exp(-5)]

    1.0000    1.0000
  148.4132    0.0067

>> B = [6;4]

      6
      4

>> ab = inv(A)*B

a    0.0267
b    5.9733
```

Plotting the optimal path

```
>> t = [0:0.01:10]';
>> x = a*exp(0.6325*t) + b*exp(-0.6325*t);
>> plot(t,x)
>> xlabel('seconds');
>> ylabel('x')
>> title('Optimal Path');
```



5) Find the function, x(t), which minimizes the following funcional

$$J = \int_0^{10} (2x^2 + 5u^2)dt$$

$$\dot{x} = -0.2x + u$$

x(0) = 6

x(10) = 7

The functional for this problem (including a LaGrange multiplier) is

$$F = (2x^2 + 5u^2) + m(\dot{x} + 0.2x - u)$$

Solving three Euler LaGrange equations

$$F_x - \tfrac{d}{dt}(F_{x'}) = 0$$

$$(4x + 0.2m) - \tfrac{d}{dt}(m) = 0$$

(1) $$4x + 0.2m - \dot{m} = 0$$

$$F_u - \tfrac{d}{dt}(F_{u'}) = 0$$

(2) $$10u - m = 0$$

$$F_m - \tfrac{d}{dt}(F_{m'}) = 0$$

(3) $$\dot{x} + 0.2x - u = 0$$

Substituting

$$m = 10u$$

$$4x + 2u - 10\dot{u} = 0$$

$$u = \dot{x} + 0.2x$$

$$\dot{u} = \ddot{x} + 0.2\dot{x}$$

$$4x + 2(\dot{x} + 0.2x) - 10(\ddot{x} + 0.2\dot{x}) = 0$$

Simplifying

$$-10\ddot{x} + 4.4x = 0$$

$$(-10s^2 + 4.4)x = 0$$

meaning

s = {0.6633, -0.6633}

and

$$x(t) = a \cdot e^{0.6633t} + b \cdot e^{-0.6633t}$$

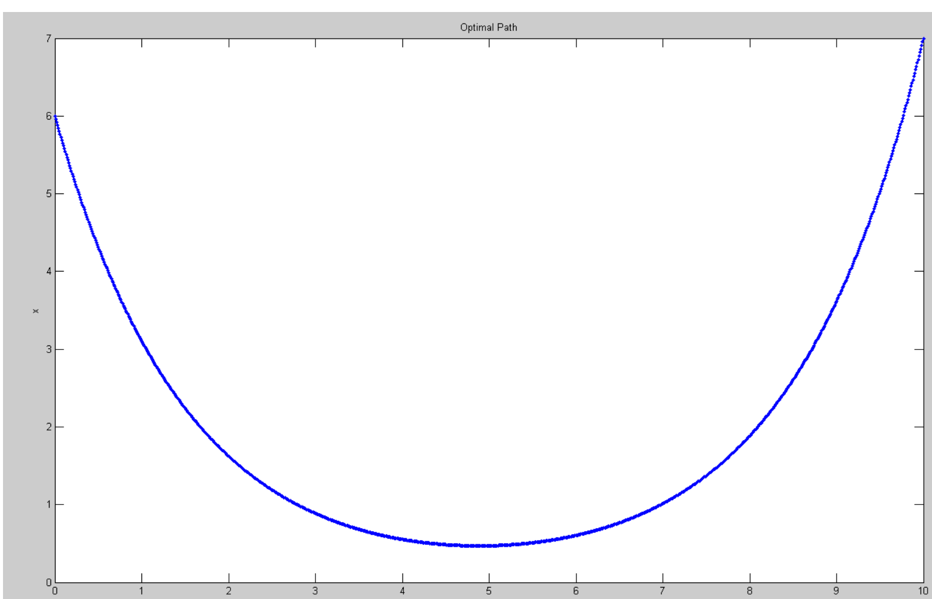
Plugging in the endpoints

$$x(0) = 6 = a + b$$

$$x(10) = 7 = a \cdot e^{6.633} + b \cdot e^{-6.633}$$

Solving

```
>> s = roots([-10,0,4.4])

    0.6633
   -0.6633

>> A = [1,1 ; exp(s(1)*10),exp(s(2)*10)]

    1.0000    1.0000
  759.9477    0.0013

>> B = [6;7];
>> ab = inv(A)*B

    0.0092
    5.9908

>> t = [0:0.01:10]';
>> a = ab(1);
>> b = ab(2);
>> x = a*exp(s(1)*t) + b*exp(s(2)*t);
>> plot(t,x);
>> xlabel('Time');
```



LQG Control

6) Cart & Pendulum (HW #4 & HW#6):

$$s\begin{bmatrix} x \\ \theta \\ \dot{x} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -29.4 & 0 & 0 \\ 0 & 26.133 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ \dot{x} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ -0.667 \end{bmatrix}F$$

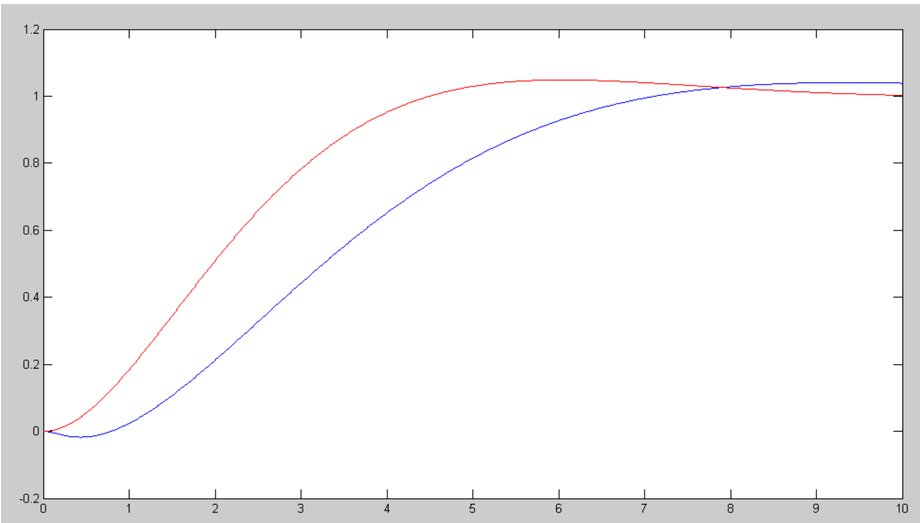
Design a full-state feedback control law of the form

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

for the cart and pendulum system from homework #4 using LQG control so that

- The DC gain is 1.00
- The 2% settling time is 8 seconds, and
- There is less than 5% overshoot for a step input.

```
>> t = [0:0.01:10]';
>> Gd = tf(0.52,[1,1,0.52])

>> A = [0,0,1,0;0,0,0,1;0,-29.4,0,0;0,26.133,0,0];
>> B = [0;0;1;-0.667];
>> C = [1,0,0,0];
>> D = 0;
>> Kx = lqr(A,B,diag([1,0,0,0]),1)

Kx =   -1.0000   -91.2225   -3.2623   -21.2933

>> DC = -C*inv(A-B*Kx)*B;
>> Kr = 1/DC;
>> G = ss(A-B*Kx,B*Kr,C,D);
>> y = step(G,t);
>> plot(t,y,'b',t,yd,'r')
```



Desired step response (red) & actual step response (blue)

Adjust Q and R until the two are close

- Increasing Q(1) speeds up the system (weighting on x)
- Increasing Q(3) adds more friction (weighting on dx)

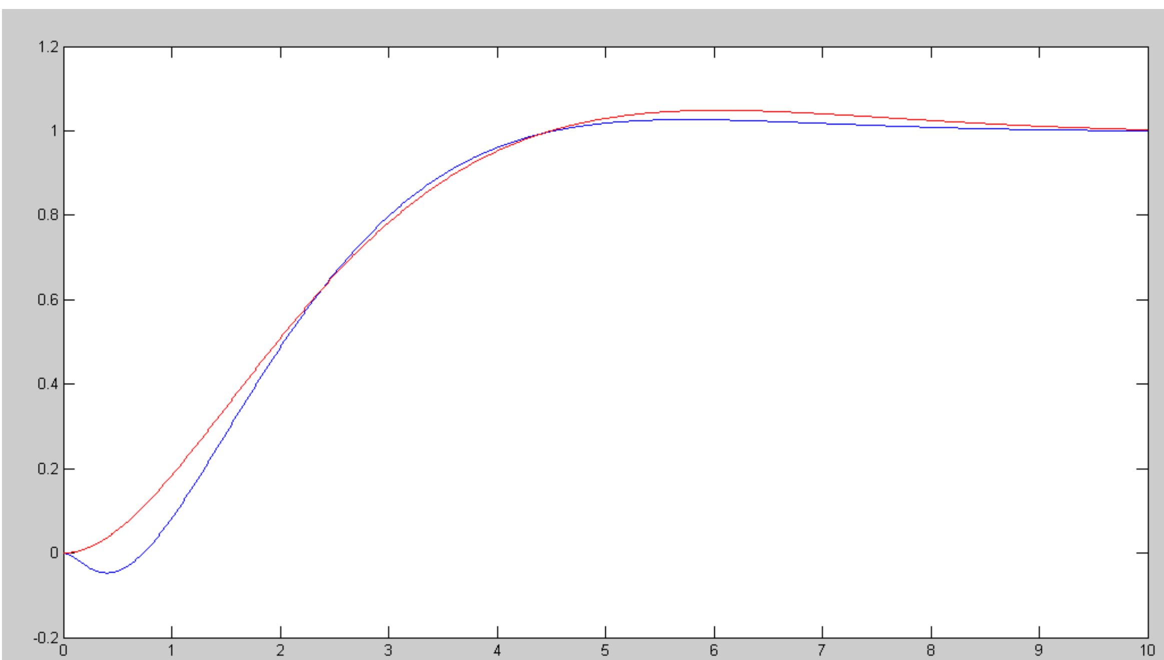
```
Kx = lqr(A,B,diag([10,0,2,0]),1);
DC = -C*inv(A-B*Kx)*B;
Kr = 1/DC;
G = ss(A-B*Kx,B*Kr,C,D);
y = step(G,t);
plot(t,y)
plot(t,y,'b',t,yd,'r')
```

The final results is

```
Kx =   -3.1623 -105.7740   -6.7131  -27.5987
Kr =   -3.1623
```

with closed-loop poles at (about the same as pole placement)

```
>> eig(A - B*Kx)

  -5.6551
  -4.6932
  -0.6735 + 0.5689i
  -0.6735 - 0.5689i
```



7) Ball and Beam (HW #4 & HW#6):

$$s\begin{bmatrix} r \\ \theta \\ \dot{r} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -7 & 0 & 0 \\ -7.84 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} r \\ \theta \\ \dot{r} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0.4 \end{bmatrix}T$$

Design a full-state feedback control law of the form

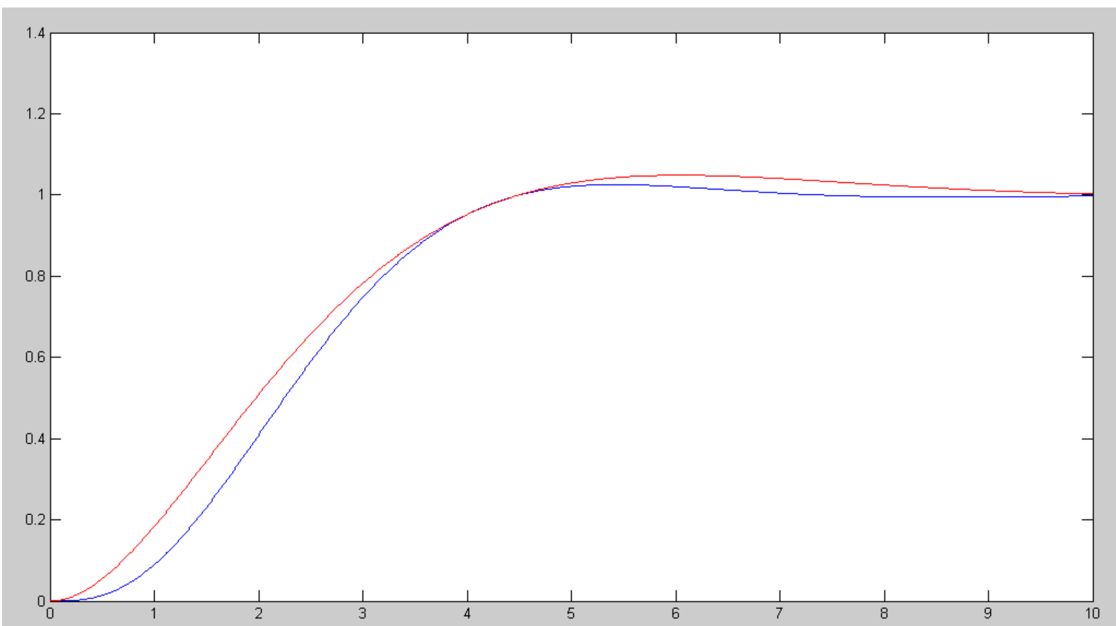
$$T = U = K_r R - K_x X$$

for the ball and beam system from homework #4 using LQG control so that

- The DC gain is 1.00
- The 2% settling time is 8 seconds, and
- There is less than 5% overshoot for a step input.

Repeating problem #6 with a new {A,B} results in

```
Kx = lqr(A,B,diag([1,0,300,20000]),1);
DC = -C*inv(A-B*Kx)*B;
Kr = 1/DC;
G = ss(A-B*Kx,B*Kr,C,D);
y = step(G,t);
plot(t,y)
plot(t,y,'b',t,yd,'r')
```



Compare your results with homework #6

- Where are the closed-loop poles with pole placement and with LQG control?
- Are the feedback gains larger or smaller with LQG control?
- Which one works better?

With LQR

```
Kx =  -39.2255  300.0466  -44.5247  146.6296

Kr =  -19.6255

>> eig(A-B*Kx)

 -56.5688
  -0.6143 + 0.8716i
  -0.6143 - 0.8716i
  -0.8544
```

With pole placement

```
>> Kx2 = ppl(A, B, [-0.5+j*0.52,-0.5-j*0.52,-3,-4])

Kx2 =  -21.8303   48.8010   -5.5867   20.0000

>> DC = -C*inv(A-B*Kx2)*B;
>> Kr2 = 1/DC

Kr2 =   -2.2303
```

Pole-Placement actually gave lower gains. If you allow the system to behave the way it wants (not trying to slow it down)

```
Kx = lqr(A,B,diag([1,0,100,100]),1);
DC = -C*inv(A-B*Kx)*B;
Kr = 1/DC;
G = ss(A-B*Kx,B*Kr,C,D);
y = step(G,t);
plot(t,y)
plot(t,y,'b',t,yd,'r')

Kx =  -39.2255   74.5176  -22.7561   21.7391

Kr =  -19.6255

>> eig(A - B*Kx)

  -4.7164
  -1.1830 + 2.4131i
  -1.1830 - 2.4131i
  -1.6132
```

I can get a faster system using similar gains using LQR