MIMO LQG Control

NDSU ECE 463/663

Lecture #26

Inst: Jake Glower

Please visit Bison Academy for corresponding
lecture notes, homework sets, and solutions

MIMO LQG Control

Bass Gura (pole placement) only works for a single input

- What do you do with multiple inputs?

Option 1:

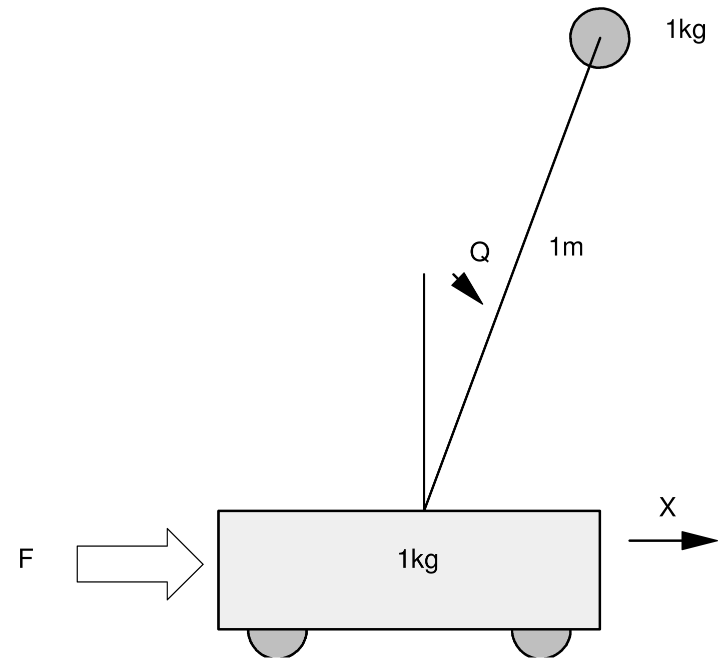
- Turn off all but one input
- Assume a constant relationship (T = 2xF)

Both are sub-optimal

Example: Cart and Pendulum

- Input 1: Force on cart
- Input 2: Torque on beam

$$s\begin{bmatrix} x \\ \theta \\ sx \\ s\theta \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -19.6 & 0 & 0 \\ 0 & 29.4 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ sx \\ s\theta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \end{bmatrix}F + \begin{bmatrix} 0 \\ 0 \\ -2 \\ 3 \end{bmatrix}T$$



Result is 8 feedback gains for 4 poles

$$\begin{bmatrix} F \\ T \end{bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} \\ k_{21} & k_{22} & k_{23} & k_{24} \end{bmatrix}\begin{bmatrix} x \\ \theta \\ sx \\ s\theta \end{bmatrix}$$

Effect of the Weightings on R:

R tells you how much cost is associated with each input.

- Changing weightings in R changes each input's contribution

Example: Find the optimal feedback gains for

$$Q = C^T C = \begin{bmatrix} 1\;0\;0\;0 \\ 0\;0\;0\;0 \\ 0\;0\;0\;0 \\ 0\;0\;0\;0 \end{bmatrix} \qquad R = \begin{bmatrix} 1\;0 \\ 0\;1 \end{bmatrix}$$

```
Q = diag([1,0,0,0]);
R = diag([1,1]);
Kx = lqr(A, B, Q, R)
```

0.8816	**-4.6251**	**2.0757**	**0.4243**	***force***
0.4720	**18.4399**	**1.3875**	**4.2661**	***torque***

Repeat for

$$R = \begin{bmatrix} 1000 & 0 \\ 0 & 1 \end{bmatrix}$$

```
>> R = diag([1000,1]);
>> Kx = lqr(A, B, Q, R)

    0.0316    0.0147    0.4352    0.2889    force
   -0.0473   19.5890    0.2619    3.7911    torque

>> eig(A - B*Kx)

  -5.4253 + 0.1843i
  -5.4253 - 0.1843i
  -0.0725 + 0.0725i
  -0.0725 - 0.0725i
```

Repeat for

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 1000 \end{bmatrix}$$

```
>> R = diag([1,1000]);
>> Kx = lqr(A, B, Q, R)
```

```
   -0.9755  -68.2469   -2.7818  -14.3206   force
    0.0070    0.2406    0.0184    0.0558   torque
```

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

  -5.4218 + 0.0618i
  -5.4218 - 0.0618i
  -0.4129 + 0.4036i
  -0.4129 - 0.4036i
```

Note:

- The closed-loop system is always stable (property of LQR)
- If you have multiple inputs, you can use any and all of them.
- You can also adjust how much control effort comes from each input

Effect on the Weightings of Q

Q penalizes the states:

- Higher weightings force the corresponding state converge faster

Example: Design a control law so that

- The cart position (x) has a 2% settling time less than 3 seconds
- With less than 5% overshoot, and
- The angle remains less than 20 degrees (0.35 radians)

Start with Q weighting the cart position (x):

$$Q = \begin{bmatrix} 1 & & & \\ & 0 & & \\ & & 0 & \\ & & & 0 \end{bmatrix} \quad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

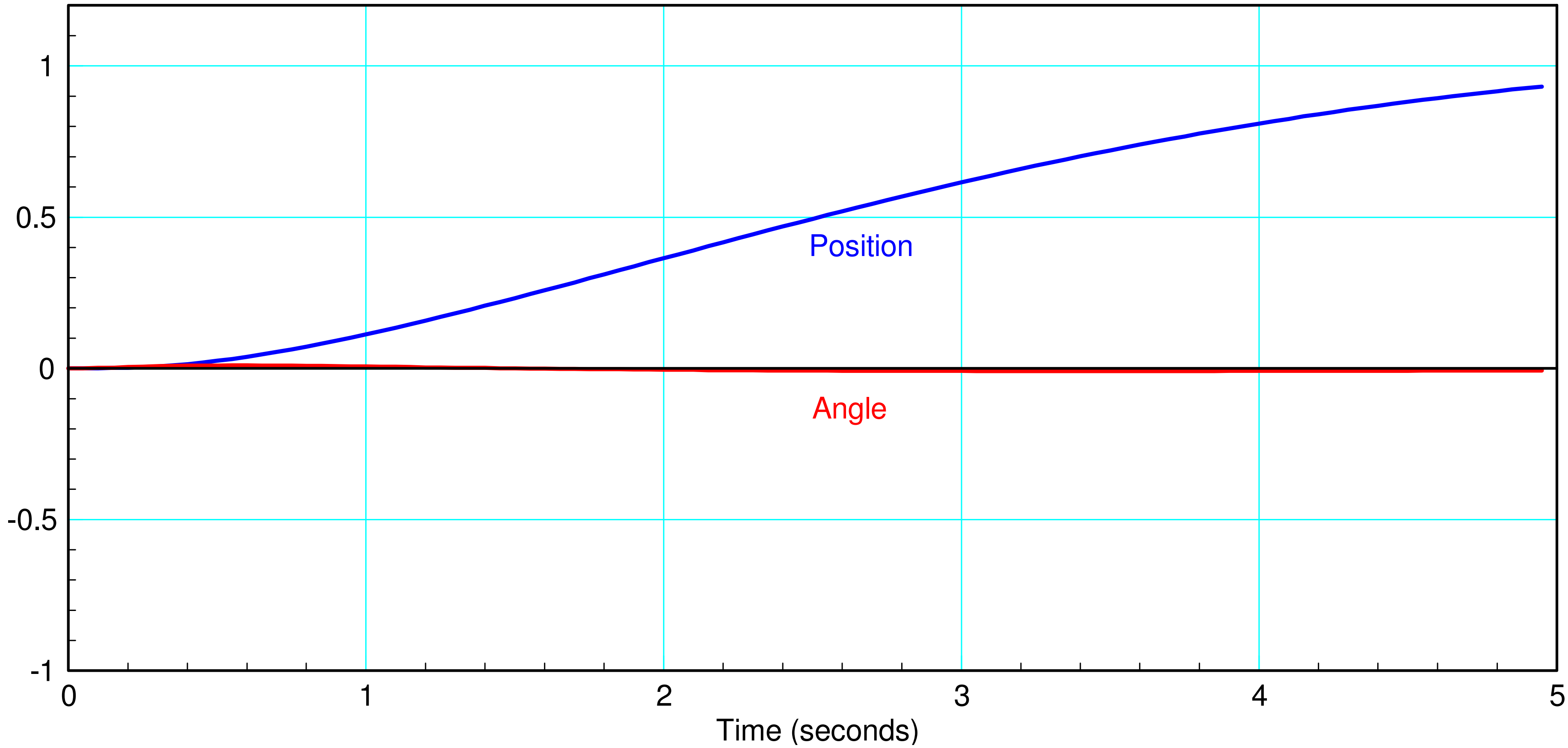
Find the feedback gains, Kx:

```
Q = diag([1,0,0,0]);
R = diag([1,1]);
Kx = lqr(A, B, Q, R)

    0.8816   -4.6251    2.0757    0.4243
    0.4720   18.4399    1.3875    4.2661
```

Set the DC gain to one:

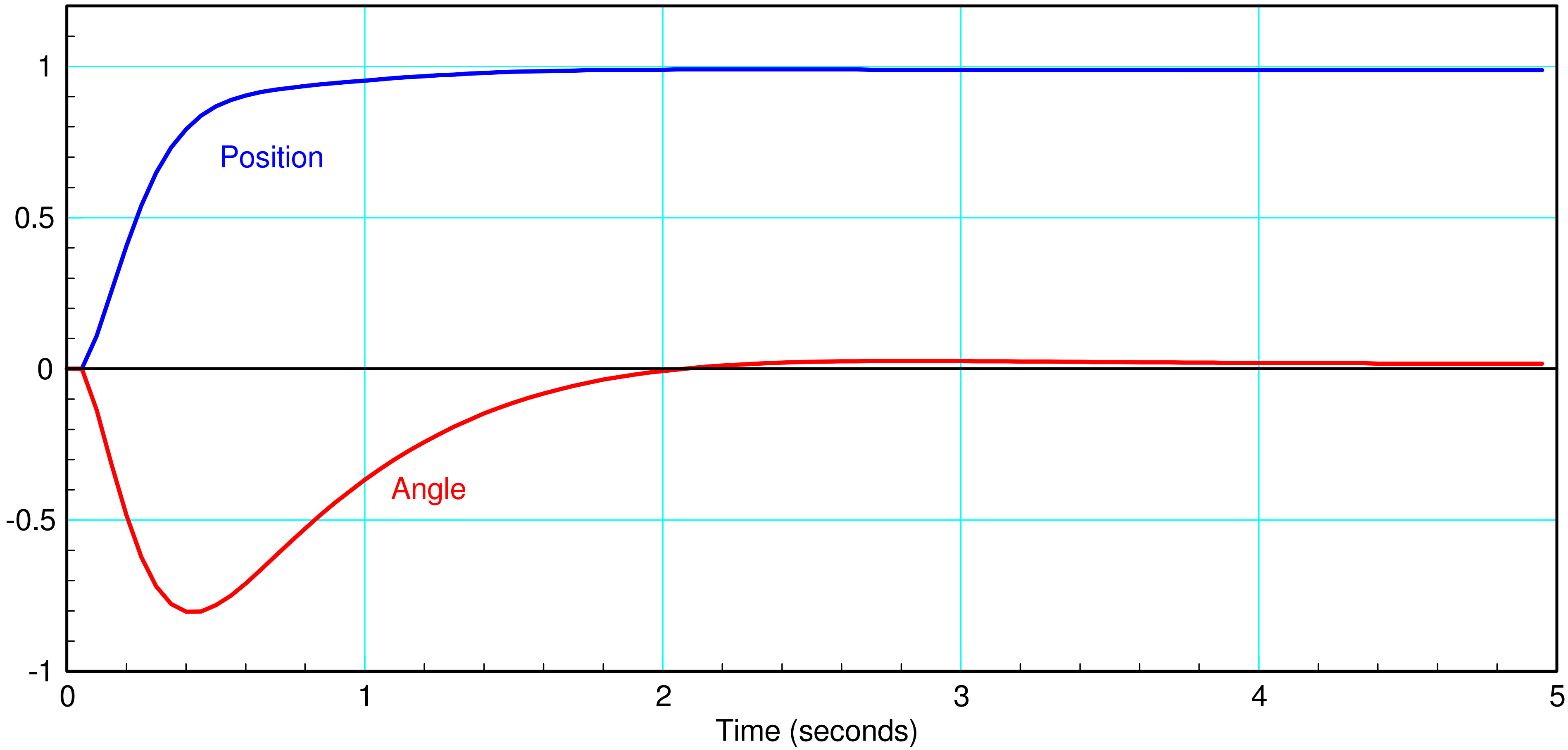
```
>> DC = -Cxq*inv(A-B*Kx)*B

    0.8816    0.4720
   -0.0482    0.0900


>> Kr = inv(DC)

    0.8816   -4.6251
    0.4720    8.6399
```

The step response from Xref is then:



This is a bit slow. Make Q larger:

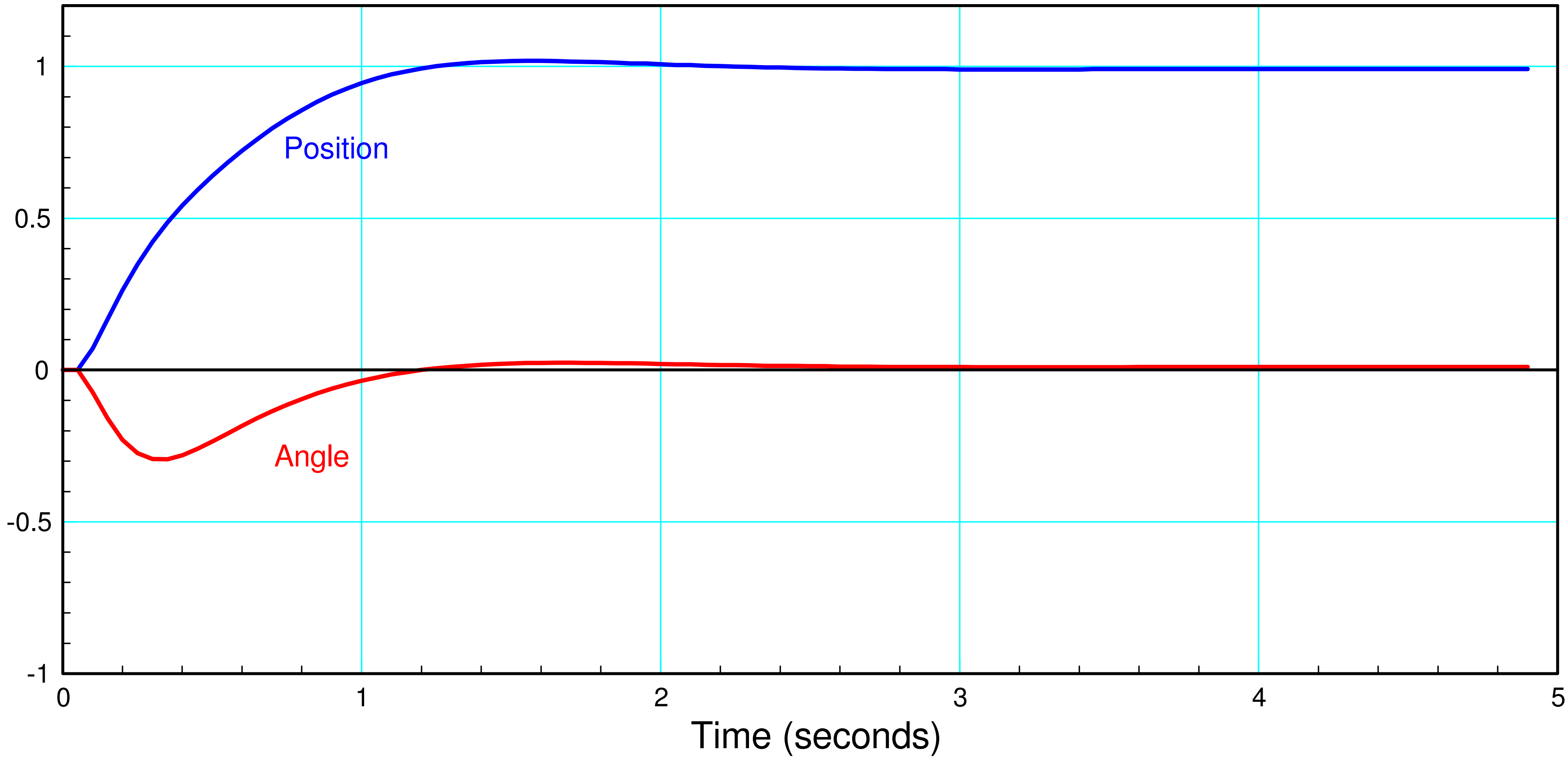
```
>> Q = diag([1000,0,0,0]);
>> R = diag([1,1]);
>> Kx = lqr(A, B, Q, R)

   25.9388    5.6056   10.9015    5.9345
  -18.0881   17.8385    0.7840    4.7834

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

    0.0259   -0.0181
    0.0584    0.0837

>> Kr = inv(DC)

   25.9388    5.6056
  -18.0881    8.0385

>> G = ss( A - B*Kx,  B*Kr,  Cxq, D);
>> y = step(G,t);
>> plot(t,y)
```



Step Response to Xref: Position (blue) and Angle (green). Q = diag(1000 0 0 0) R = diag(1 1)

To reduce the angle, increase the weighting on the second state (angle).

```
>> Q = diag([1000,1000,0,0]);
>> R = diag([1,1]);
>> Kx = lqr(A, B, Q, R)

   31.1386    5.7709   12.8486    7.1439
   -5.5123   42.3996    3.2744    7.5397

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

    0.0311   -0.0055
    0.0053    0.0297

>> Kr = inv(DC)

   31.1386    5.7709
   -5.5123   32.5996
```

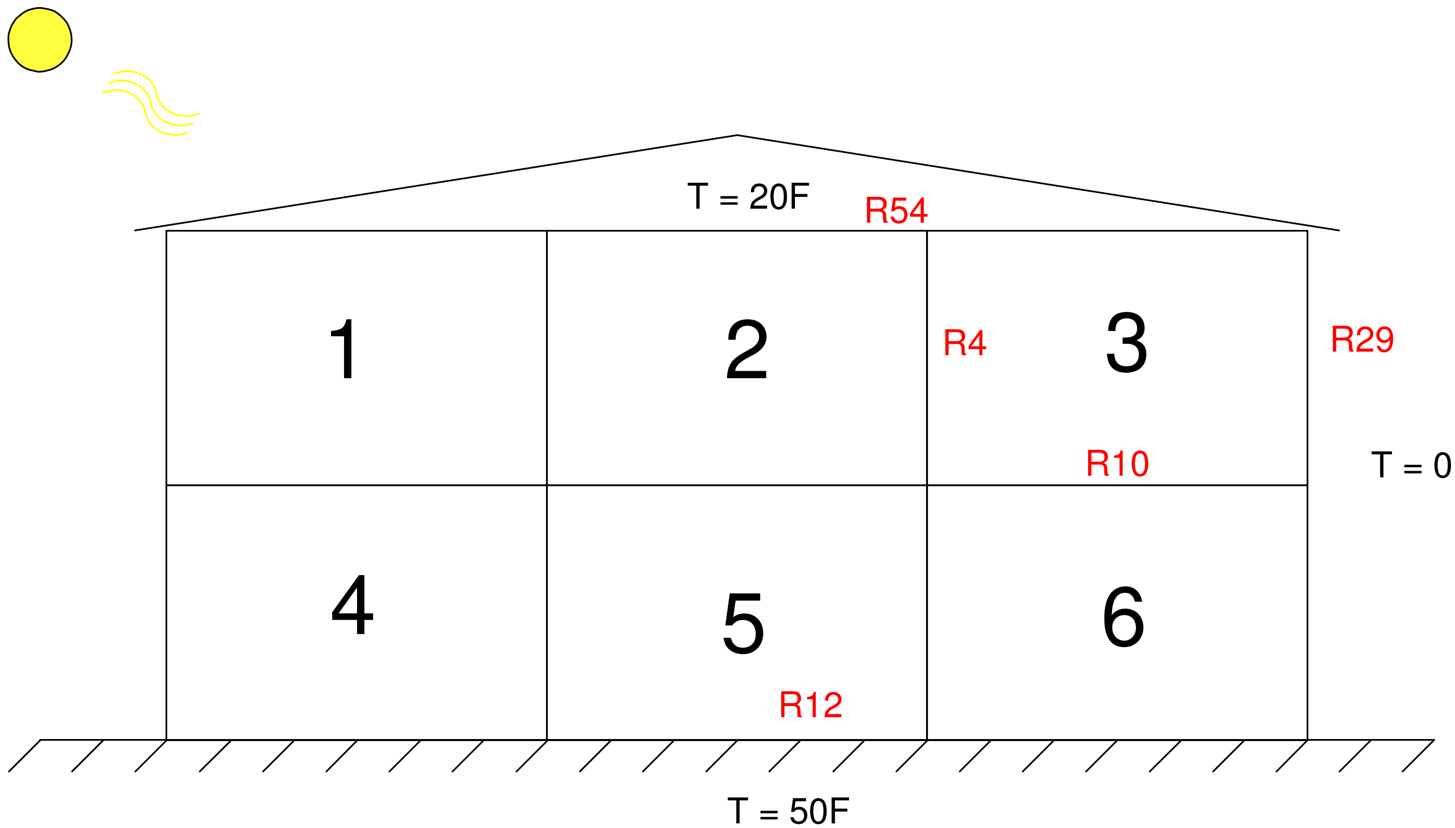


Step Response to Xref: Position (blue) and Angle (green). Q = diag(1000 1000 0 0) R = diag(1 1)

Now the system is reasonable:

- The position reaches its final value in about 3 seconds
- With 2.6% overshoot
- The maximum angle is -0.3 radians (17 degrees)

Example 2: Apartment Complex



Dynamics:

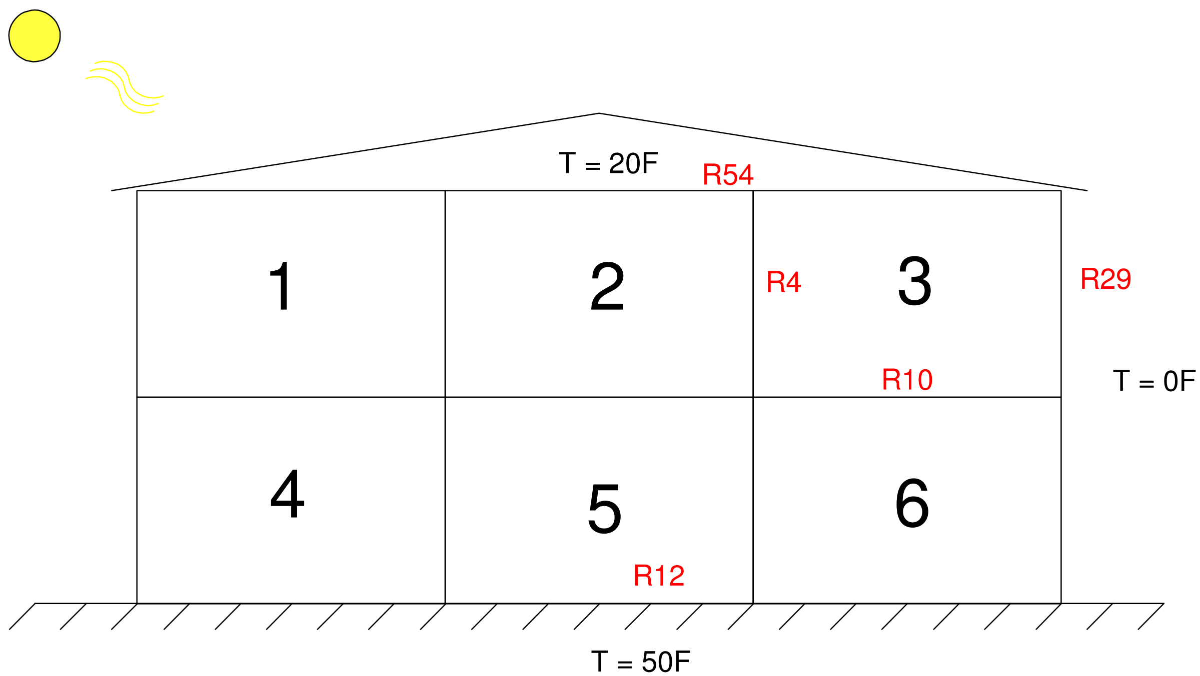
How to use all six inputs?

$$sX = \begin{bmatrix} -0.403 & 0.25 & 0 & 0.1 & 0 & 0 \\ 0.25 & -0.618 & 0.25 & 0 & 0.1 & 0 \\ 0 & 0.25 & -0.403 & 0 & 0 & 0.1 \\ 0.1 & 0 & 0 & -0.468 & 0.25 & 0 \\ 0 & 0.1 & 0 & 0.25 & -0.683 & 0.25 \\ 0 & 0 & 0.1 & 0 & 0.25 & -0.468 \end{bmatrix} X + \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \end{bmatrix} + \begin{bmatrix} 0.37 \\ 0.37 \\ 0.37 \\ 4.1 \\ 4.1 \\ 4.1 \end{bmatrix}$$

If you turn off the heat, what temperature do the rooms converge to?

inv(A)*B_{dist}

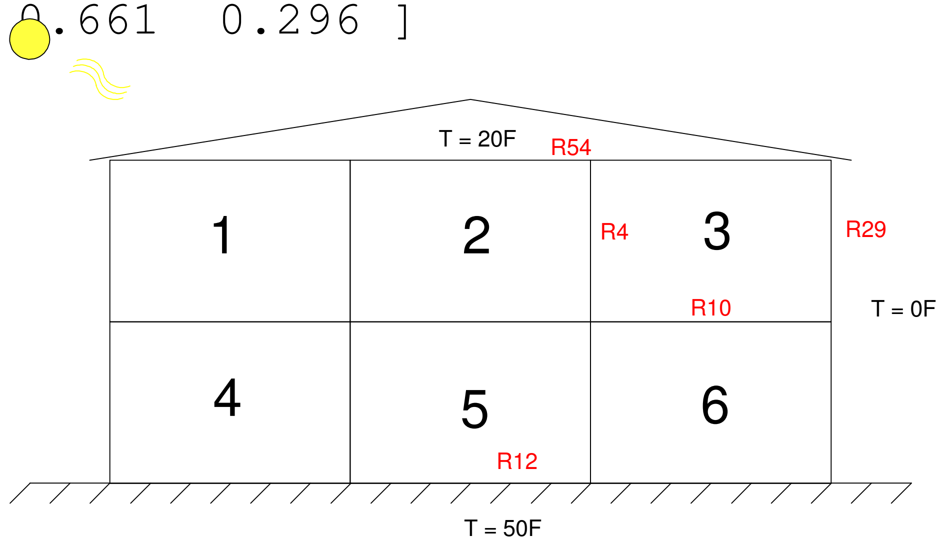
T1	25.143493
T2	26.309411
T3	25.143493
T4	31.854749
T5	33.174694
T6	31.854749



Option 1: Turn off all inputs save #5

$$sX = \begin{bmatrix} -0.403 & 0.25 & 0 & 0.1 & 0 & 0 \\ 0.25 & -0.618 & 0.25 & 0 & 0.1 & 0 \\ 0 & 0.25 & -0.403 & 0 & 0 & 0.1 \\ 0.1 & 0 & 0 & -0.468 & 0.25 & 0 \\ 0 & 0.1 & 0 & 0.25 & -0.683 & 0.25 \\ 0 & 0 & 0.1 & 0 & 0.25 & -0.468 \end{bmatrix} X + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} U_5 + \begin{bmatrix} 0.37 \\ 0.37 \\ 0.37 \\ 4.1 \\ 4.1 \\ 4.1 \end{bmatrix}$$

```
Kx   = [ 0.189   0.227  0.189  0.296  0.661  0.296 ]

T1     58.022489
T2     63.076242
T3     58.022489
T4     72.440025
```
T5   95.998731
```
T6     72.440025
```

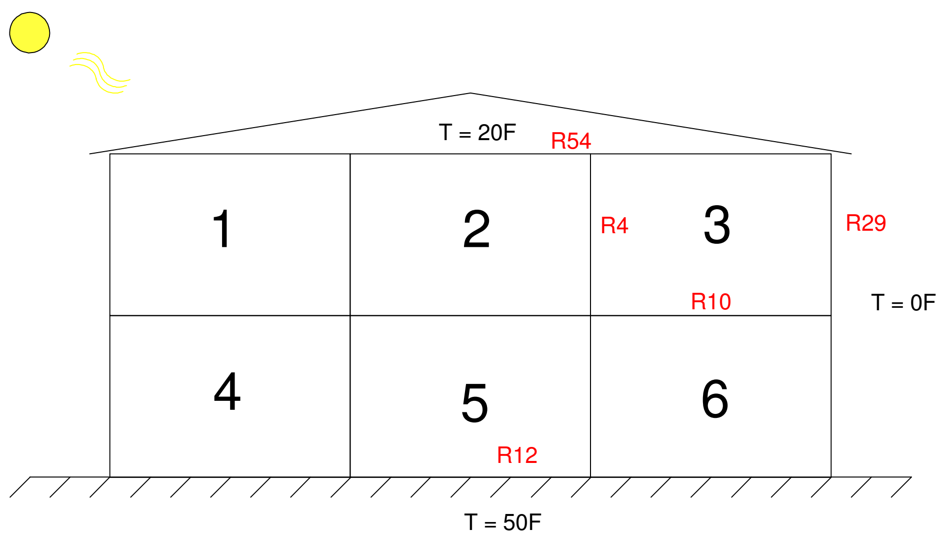


Option 2: Make all heaters output identical heat

$$sX = \begin{bmatrix} -0.403 & 0.25 & 0 & 0.1 & 0 & 0 \\ 0.25 & -0.618 & 0.25 & 0 & 0.1 & 0 \\ 0 & 0.25 & -0.403 & 0 & 0 & 0.1 \\ 0.1 & 0 & 0 & -0.468 & 0.25 & 0 \\ 0 & 0.1 & 0 & 0.25 & -0.683 & 0.25 \\ 0 & 0 & 0.1 & 0 & 0.25 & -0.468 \end{bmatrix} X + \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} U + \begin{bmatrix} 0.37 \\ 0.37 \\ 0.37 \\ 4.1 \\ 4.1 \\ 4.1 \end{bmatrix}$$

```
-inv(A - B*Kx)*(B*Kr*(70-Zero) + Dist)
```

```
T1      70.444307
T2      73.615864
T3      70.444307
T4      67.540638
T5      70.414247
T6      67.540638
```

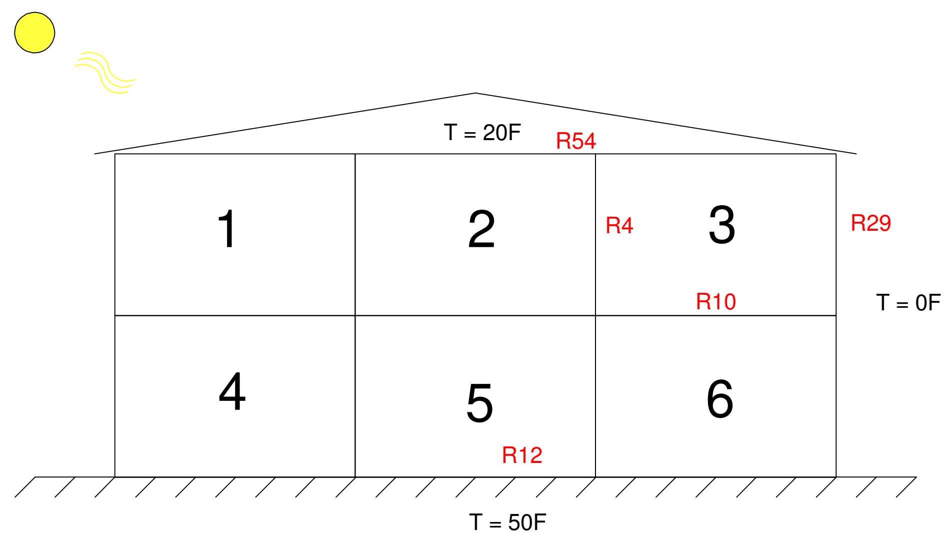


Option 3: Adjust the heat output

$$sX = \begin{bmatrix} -0.403 & 0.25 & 0 & 0.1 & 0 & 0 \\ 0.25 & -0.618 & 0.25 & 0 & 0.1 & 0 \\ 0 & 0.25 & -0.403 & 0 & 0 & 0.1 \\ 0.1 & 0 & 0 & -0.468 & 0.25 & 0 \\ 0 & 0.1 & 0 & 0.25 & -0.683 & 0.25 \\ 0 & 0 & 0.1 & 0 & 0.25 & -0.468 \end{bmatrix} X + \begin{bmatrix} 1 \\ 0.4 \\ 1 \\ 1.4 \\ 0.5 \\ 1.4 \end{bmatrix} U + \begin{bmatrix} 0.37 \\ 0.37 \\ 0.37 \\ 4.1 \\ 4.1 \\ 4.1 \end{bmatrix}$$

```
-inv(A - B*Kx)*(B*Kr*(70-Zero) + Dist)
```

```
T1     69.603963
T2     70.240779
T3     69.603963
T4     70.276628
T5     69.99804
T6     70.276628
```



Treat all inputs separately:

- LQR with $B = I_{6x6}$

```
-inv(A - B*Kx)*(B*Kr*(70-Zero) + Dist)

    68.511245
    69.751752
    68.511245
    70.630332
    71.965095
    70.630332

Kx

    0.701    0.141    0.022    0.062    0.017    0.002
    0.141    0.602    0.141    0.017    0.050    0.017
    0.022    0.141    0.701    0.002    0.017    0.062
    0.062    0.017    0.002    0.661    0.130    0.021
    0.017    0.050    0.017    0.130    0.570    0.130
    0.002    0.017    0.062    0.021    0.130    0.661
```

Plus:

- Much better control
- Each apt is regulated to 70F (approx)

Minus:

- Each apartment must know the temperature of each other apartment

Problem:

- Can you just use local information?
- Can you constrain Kx to be diagonal?

LQR does not allow this: you get full-state feedback gains