Lecture 15

ECE/ME 179D

1. LQR: Linear Quadratic Regulator

- System dynamics are LINEAR
- Cost function is QUADRATIC
- Controller REGULATES states, i.e., drives them to zero (and not to a reference trajectory)

Topics:
1. LQR
2. "≈rel" versus "≈abs", (again...)

- LQR approach sets the values in the controller gain matrix, K, for the control law:

$$\underset{(m\times 1)}{u} = -\underset{(m\times n)}{K}\underset{(n\times 1)}{x}$$

n = # of states
m = # of inputs, (e.g., torques)

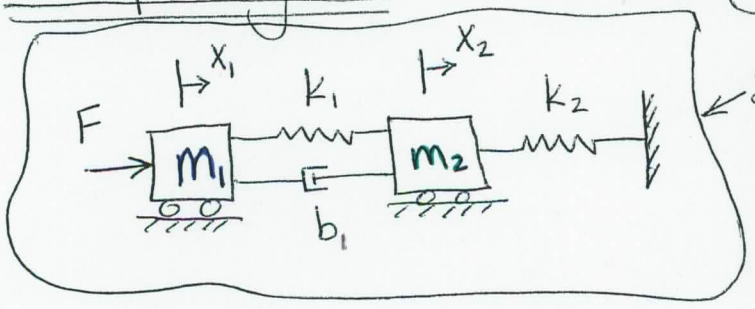
Such that the quadratic cost function below is MINIMIZED:

(of course, "x" is "x(t)", "u" is "u(t)", i.e., they are varying w/ time)

$$J = \int_{t=0}^{\infty} \left\{ \underset{(1\times n)(n\times n)(n\times 1)}{x^T Q x} + \underset{(1\times m)(m\times m)(m\times 1)}{u^T R u} \right\}$$

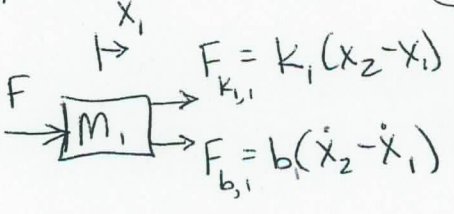
- Both Q & R are symmetric, positive definite.
- Often, Q & R are simply diagonal matrices, in practice.
- When m=1 (one actuator), R is 1×1, so: $u^T R u = R u^2$

Example System: Two-mass "spring-mass-damper"...



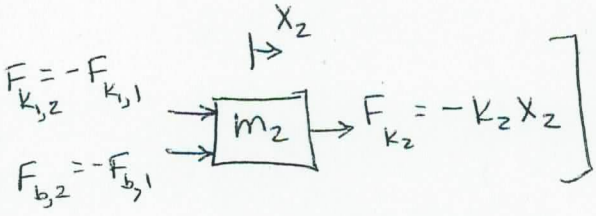
See also MATLAB code

By inspection of a free-body diagram for each mass:



EOM 1

$$m_1\ddot{x}_1 = F + k_1(x_2 - x_1) + b_1(\dot{x}_2 - \dot{x}_1)$$



EOM 2

$$m_2\ddot{x}_2 = -k_1(x_2 - x_1) - b_1(\dot{x}_2 - \dot{x}_1) - k_2 x_2$$

Using absolute coordinates for x_1 & x_2, let's define the state vector, X, as:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \dot{x}_1 \\ \dot{x}_2 \end{bmatrix},$$

Then the EOMs in matrix form are:

$$\dot{X} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \ddot{x}_1 \\ \ddot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{k_1}{m_1} & \frac{k_1}{m_1} & -\frac{b_1}{m_1} & \frac{b_1}{m_1} \\ \frac{k_1}{m_2} & -\frac{(k_1+k_2)}{m_2} & \frac{b_1}{m_2} & -\frac{b_1}{m_2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{m_1} \\ 0 \end{bmatrix} F$$

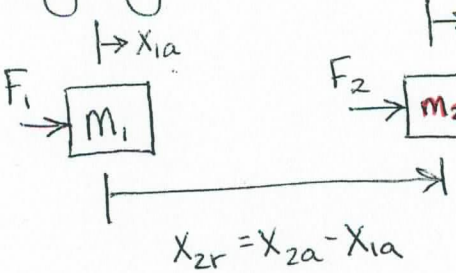
$$\dot{X} = A \quad X + B \quad u$$

"$\dot{X} = AX + Bu$"

(See MATLAB code for example that creates a state space system & uses LQR for feedback control...)

---

2. Let's also use a simplified two-mass system to better understand how to determine "$\tilde{=}$" for the Lagrangian EOMs...



← Systems are completely decoupled:

$m_1 \ddot{x}_{1a} = F_1$

$m_2 \ddot{x}_{2a} = F_2$

See also "L12 Supplement" (Lecture 12)

• Lagrangian approach:

GCs?	x_{1a}(absolute), x_{2a}(abs)	x_{1a}(abs), x_{2r}(rel)
T^*?	$T^* = \frac{1}{2} m_1 \dot{x}_{1a}^2 + \frac{1}{2} m_2 \dot{x}_{2a}^2$	$T^* = \frac{1}{2} m_1 \dot{x}_{1a}^2 + \frac{1}{2} m_2 (\dot{x}_{1a} + \dot{x}_{2r})^2$
V ?	$V = 0$	$V = 0$
$\mathcal{L} = T^* - V$	$\mathcal{L} = T^*$	$\mathcal{L} = T^*$

EOM 2:

$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{x}_{2a}}\right) - \frac{\partial \mathcal{L}}{\partial x_{2a}} = \tilde{=}_{2a}$ → (2A) → $m_2 \ddot{x}_{2a} = F_2$

$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{x}_{2r}}\right) - \frac{\partial \mathcal{L}}{\partial x_{2r}} = \tilde{=}_{2r}$ → (2R) → $m_2(\ddot{x}_{1a} + \ddot{x}_{2r}) = F_2$

LHS's are the same, ∴ RHS's must also match.

EOM 1:

$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{x}_{1a}}\right) - \frac{\partial \mathcal{L}}{\partial x_{1a}} = \tilde{=}_{1a}$ → (1A) → $m_1 \ddot{x}_{1a} = F_1$

$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{x}_{1a}}\right) - \frac{\partial \mathcal{L}}{\partial x_{1a}} = \tilde{=}_{1r}$

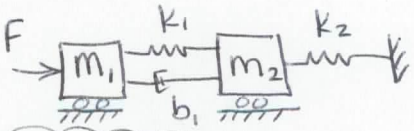
$m_1 \ddot{x}_{1a} + m_2(\ddot{x}_{1a} + \ddot{x}_{2r}) = \underline{?} = \tilde{=}_{1r}$

$\left(\text{LHS of } (1A)\right) + \left(\text{LHS of } (2A)\right) = \left(\text{RHS } (1A) + \text{RHS } (2A)\right)$ (3)

$\therefore \tilde{=}_{1r} = \tilde{=}_{1a} + \tilde{=}_{2a} = F_1 + F_2$

RHS (1A) RHS (2A)

Matlab m-file "sample_lqr.m"




```
1 clear all
2 % This code models a two-cart system, as describe in class
3 % during Lecture 15. A force is input at m1. m1 and m2
4 % are connected by k1 and b1, and k2 connects m2 to a wall.
5 % Katie Byl, 2012.
6
7 m1 = 5; m2 = 2;
8 k1 = 200; b1 = 10; k2 = 100;
9
10 A = [0 0 1 0
11     0 0 0 1
12     -k1/m1 k1/m1 -b1/m1 b1/m1
13     k1/m2 -(k1+k2)/m2 b1/m2 -b1/m2];
14 B = [0 0 1/m1 0]'
15 C = diag([1 1 1 1]);
16 D = 0;
17
18 % Commands below create a "state space system"
19 ss1 = ss(A,B,C,D)
20
21 % Open loop poles are the eigenvalues of matrix A
22 open_loop_poles = eig(A)
23
24 %=== Code below finds feedback gains "K" via LQR (linear
25 %    quadratic regulator) approach from Lecture 15. ===
26
27 % Q gives penalties on STATE (squared errors), and
28 % R gives a penalty on the square of the control effort, u.
29 % (i.e., J = x'*Q*x + u'*R*u, from Lecture 15.)
30 % Usually, Q has more than one non-zero diagonal entry, but
31 % note it still works fine as shown below. In general, Q will
32 % be diagonal (with no off-diagonal terms), however:
33 Q = diag([1 0 0 0]);
34 R = .001;
35
36 % MATLAB can calculate the optimal gain matrix, K, for the
37 % LQR problem defined by the system dynamics (A and B matrices)
38 % and your costs, Q and R:
39 K = lqr(A,B,Q,R);
40
41 % Below is the CLOSE-LOOP system, using u=-K*x as a control law:
42 ss2 = ss(A-B*K,B,C,D);
43
44
45 % ==== We will then compare the above system to one with a new R ====
46
47 Qnew = Q;
48 % Below, let us change ONLY the value of R (with same Q):
49 Rnew = 1;
50 Knew = lqr(A,B,Qnew,Rnew);
51 % Define a TIME vector, to simulate the response, using MATLAB:
52 dt = 0.001;
53 t = 0:dt:50;
54 % Below, we also define an "initial condition" for the system:
55 X0 = [0;0;1;0];
56
57 % === Now, we will do LOTS of plots, both of STATES and of
58 %     control effort, u
59 figure(11); clf
60 % "initial" simulated the initial condition response, given X0.
61 y1 = initial(ss1,X0,t);
62 for n=1:4
63     subplot(5,1,n)
64     plot(t,y1(:,n),'b-');
65 end
66 u1 = 0*t;  % Free response has "u=0", always...
67 subplot(515)
68 plot(t,u1,'r-');
69 subplot(511)
70 title('Free Response (u=0)')
71
72 J1terms(5) = 0; % u1=0
73 for n=1:4
74     J1terms(n) = dt*sum(y1(:,n).^2);
```

"$\dot{X} = Ax + Bu$" ← Defines EOMs.

"$y = Cx + Du$" ← Defines measured outputs, y.

Type "help ss" in Matlab for usage.

Open-loop poles.

Q & R, for LQR.

"help lqr" for usage... Sets "K" gains.

Closed-loop system, w/ additional disturbance input "Bu_{dist}" $u_{tot} = u + u_{dist}$

$u_{tot} = -Kx + u_{dist}$, to allow step input in u_{dist}, closed loop.

New R & corresponding Knew

Start w/ $X_1 = X_2 = 0$, $\dot{X}_1 = 1$, $\dot{X}_2 = 0$

initial condition response

free response, $u=0$

```
end
fprintf('sqrd er:  X(1)      X(2)      X(3)      X(4)       u\n')
fprintf('u=0: %8.3f  %8.3f  %8.3f  %8.3f    %8.3f  \n',J1terms)

input('\n\nNow, let''s use LQR feedback control:')
figure(12); clf
y2 = initial(ss2,X0,t);

% Closed-loop poles are eigenvalues of "A-BK", for u=-K*x
closed_loop_poles1 = eig(A-B*K)

for n=1:4
    subplot(5,1,n)
    plot(t,y2(:,n),'b-');
    hold on
end
subplot(515)

% Line below calculates the CONTROL EFFORT, u:
u2 = -y2*K';

hold on
plot(t,u2,'r-'); % u is the actuation, assuming "u=-K*x" is the law
J2terms(5) = dt*sum(u2.^2);
for n=1:4
    J2terms(n) = dt*sum(y2(:,n).^2);
end
fprintf('\n');
fprintf('C1:  Q1:%5.3f  Q2:%5.3f  Q3:%5.3f  Q4:%5.3f    R:%5.3f  \n',...
    Q(1,1),Q(2,2),Q(3,3),Q(4,4),R(1))
fprintf('C1:  %8.3f  %8.3f  %8.3f  %8.3f    %8.3f  \n',J2terms)
J2add = J2terms.*[diag(Q)', R];
fprintf('J1 = %8.3f + %8.3f + %8.3f + %8.3f   + %8.3f = %12.8f \n',...
    J2add, sum(J2add));


% Set to 1 or 0, below, to tweak vs redefine R and Q...
if 0
    % try "tweaking" K:
    input('Now, let''s TWEAK the values in K:')
    Kfac = .7+.6*rand(1,4)
    Knew = Kfac.*K;
    Qnew = Q; Rnew = R;
else
    input('\n\nNow, let''s try LQR with same Q but larger R penalty:')
    Rnew
end

% Closed loop poles with Qnew and Rnew, below:
closed_loop_poles2 = eig(A-B*Knew)

figure(12); %clf
ss3 = ss(A-B*Knew,B,C,D);
y3 = initial(ss3,X0,t);
for n=1:4
    subplot(5,1,n)
    plot(t,y3(:,n),'k-');
    Ax = axis;
    axis([0 15 Ax(3:4)])
end
subplot(515)
u3 = -y3*Knew';
plot(t,u3,'k-');
Ax = axis;
axis([0 15 Ax(3:4)])
subplot(511)
title('u = -B*K, with K set via LQR')

J3terms(5) = dt*sum(u3.^2);
for n=1:4
    J3terms(n) = dt*sum(y3(:,n).^2);
end
fprintf('\n');
fprintf('C2:  Q1:%5.3f  Q2:%5.3f  Q3:%5.3f  Q4:%5.3f    R:%5.3f  \n',...
```

Repeat initial condition response for closed-loop system, using K.

closed-loop poles

$u = -Kx$

u^2, summed over time

x^2

$J = \int_0^\infty x^T Q x + u^T R u$

$J \approx \sum_{n=1}^{N} \left(Q(1,1)x_1^2 + Q(2,2)x_2^2 + \dots + Q(3,3)\dot{x}_1^2 + Q(4,4)\dot{x}_2^2 + Ru^2 \right) * \Delta t$

(Q is diagonal, & we approximate the continuous-time integral w/ a sum.)

to test cost J when K is "tweaked" around its optimal value

CL poles, using Knew

CL system with Knew

$u = -K_{new}x$

```
149     Qnew(1,1),Qnew(2,2),Qnew(3,3),Qnew(4,4),Rnew(1))
150 fprintf('C2:  %8.3f   %8.3f   %8.3f   %8.3f     %8.3f  \n',J3terms)
151 J3add = J3terms.*[diag(Qnew)', Rnew];
152 fprintf('J2 = %8.3f + %8.3f + %8.3f + %8.3f   + %8.3f = %12.8f \n',...
153     J3add, sum(J3add));
154
155 % Now, try pole placement
156 pdes = [-20 -30 -100 -200]
157 Kplace = acker(A,B,pdes)
158 ss4 = ss(A-B*Kplace,B,C,D);
159 y4 = initial(ss4,X0,t);
160 u4 = y4*Kplace';
161
162 figure(13); clf
163 for n=1:4
164     subplot(5,1,n)
165     plot(t,y4(:,n),'b-');
166     Ax = axis;
167     axis([0 .5 Ax(3:4)])
168 end
169 subplot(515)
170 plot(t,u4,'r-');
171 Ax = axis;
172 axis([0 .5 Ax(3:4)])
173 subplot(511)
174 title('u = -B*K, with K set via Pole Placement')
175
176 J4terms(5) = dt*sum(u4.^2);
177 for n=1:4
178     J4terms(n) = dt*sum(y4(:,n).^2);
179 end
180 fprintf('\n');
181 fprintf('\n');
182 fprintf('C2:  Q1:%5.3f   Q2:%5.3f   Q3:%5.3f   Q4:%5.3f     R:%5.3f  \n',...
183     Qnew(1,1),Qnew(2,2),Qnew(3,3),Qnew(4,4),Rnew(1))
184 fprintf('C2:  %8.3f   %8.3f   %8.3f   %8.3f     %8.3f  \n',J4terms)
185 J4add = J4terms.*[diag(Qnew)', Rnew];
186 fprintf('Jp = %8.3f + %8.3f + %8.3f + %8.3f   + %8.3f = %12.8f \n',...
187     J4add, sum(J4add));
188
189
190
```

acker or place

pole placement:

K is now set (as needed) to place the 4 poles at (arbitrary), but stable, locations on the S-plane.

(See Workspace output & plots, next pages.)

Lecture 15

ECE/ME 179D

MATLAB output:

```
ss1 =
 a =
       x1   x2   x3   x4
  x1    0    0    1    0
  x2    0    0    0    1
  x3  -40   40   -2    2
  x4  100 -150    5   -5

 b =
       u1
  x1    0
  x2    0
  x3  0.2
  x4    0

 c =
       x1  x2  x3  x4
  y1    1   0   0   0
  y2    0   1   0   0
  y3    0   0   1   0
  y4    0   0   0   1

 d =
       u1
  y1    0
  y2    0
  y3    0
  y4    0

Continuous-time state-space model.

open_loop_poles =
  -3.4361 +12.8869i
  -3.4361 -12.8869i
  -0.0639 + 3.3525i
  -0.0639 - 3.3525i

Q =
     1     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
R =
   1.0000e-03
sqrd er:  X(1)      X(2)      X(3)      X(4)      u
u=0:    0.240     0.126     2.696     1.448     0.000

Now, let's use LQR feedback control:
closed_loop_poles1 =
  -3.4367 +12.8855i
  -3.4367 -12.8855i
  -0.7656 + 3.4439i
  -0.7656 - 3.4439i

C1:  Q1:1.000   Q2:0.000   Q3:0.000   Q4:0.000     R:0.001
C1:     0.018      0.010      0.226      0.145      17.045
J1 =    0.018 +    0.000 +    0.000 +    0.000  +   0.017 =   0.03513940

Now, let's try LQR with same Q but larger R penalty:
Rnew =
     1
closed_loop_poles2 =
  -3.4361 +12.8869i
  -3.4361 -12.8869i
  -0.0685 + 3.3526i
  -0.0685 - 3.3526i

C2:  Q1:1.000   Q2:0.000   Q3:0.000   Q4:0.000     R:1.000
C2:     0.224      0.118      2.515      1.352      0.008
J2 =    0.224 +    0.000 +    0.000 +    0.000  +   0.008 =   0.23157950
pdes =
   -20   -30  -100  -200

Kplace =
      32300      551450        1715       27235

C2:  Q1:1.000   Q2:0.000   Q3:0.000   Q4:0.000     R:1.000
C2:     0.000      0.000      0.002      0.000   7183.991
Jp =    0.000 +    0.000 +    0.000 +    0.000  + 7183.991 = 7183.99131103
```

$A = \begin{bmatrix} \cdots \\ \cdots \\ \cdots \\ \cdots \end{bmatrix}$

$B = \begin{bmatrix} \cdots \\ \cdots \\ \cdots \\ \cdots \end{bmatrix}$

C defines outputs

D is often all zeros.

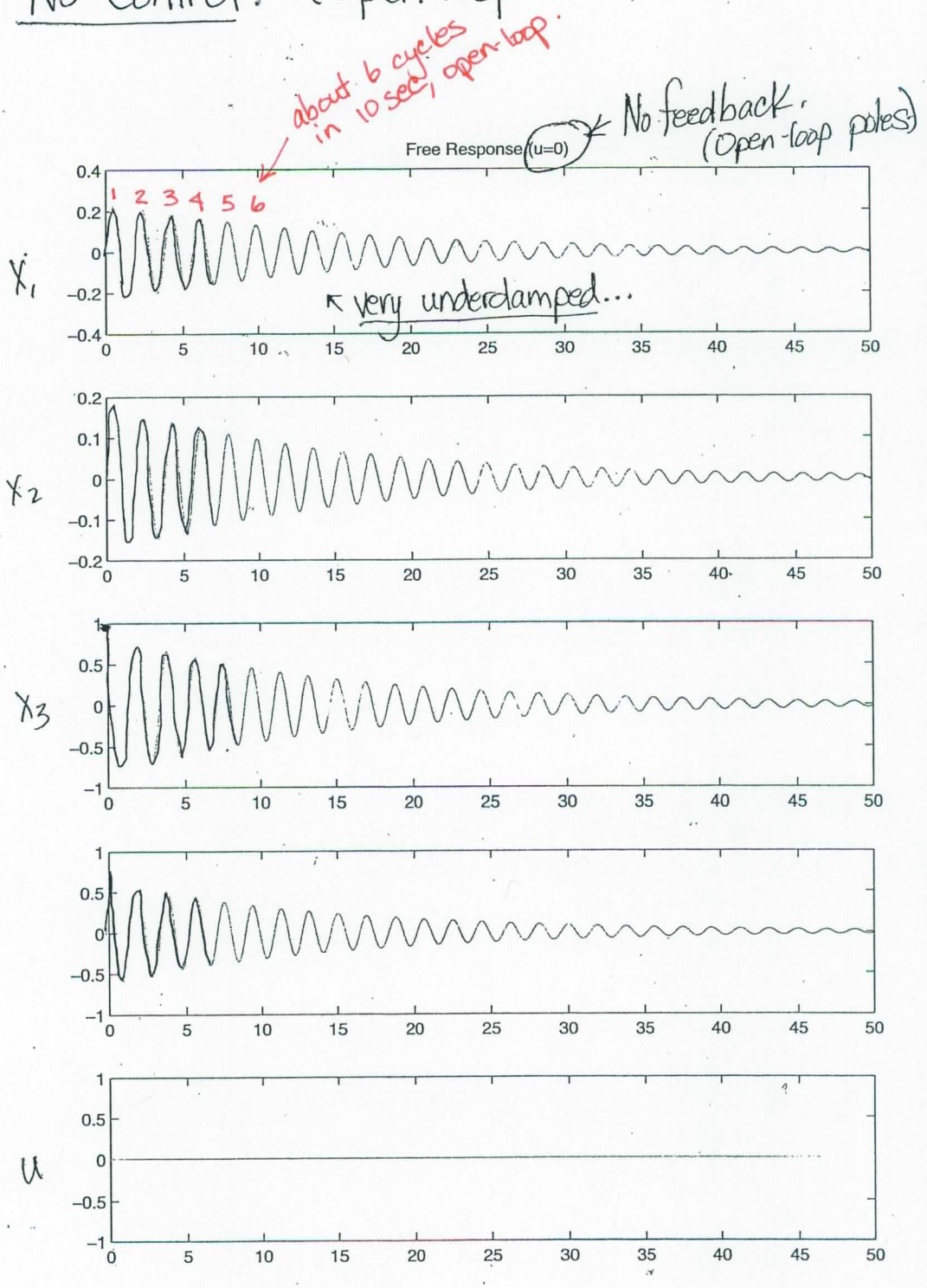
← OL poles

for LQR → $K_1 = [8.8, -2.5, 7.0, 2.0]$ ← for $R = 0.001$

Increasing penalty on "R" control effort ($u^T R u$) decreases control gains, typically.

w/ $R = 1.0$ → $K_2 = [0.026, -0.027, 0.046, 0.013]$

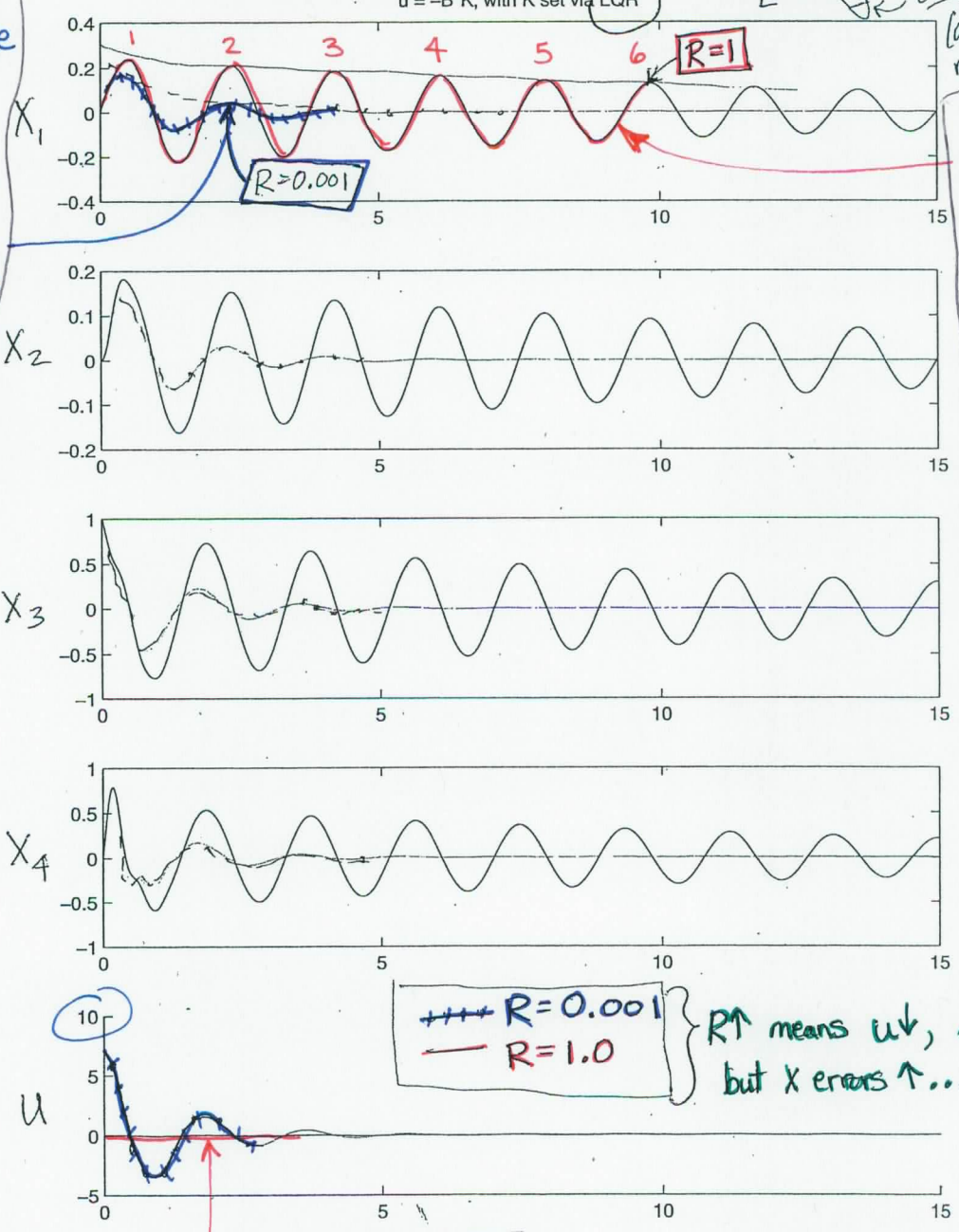
No control. (Open-loop)



LQR control, to determine K (the gain matrix).

More typically, all diagonal elements of Q have some non-zero value... $Q = \begin{bmatrix} \cdot & & & \\ & \cdot & & \\ & & \cdot & \\ & & & \cdot \end{bmatrix}$ ...& Q is often diagonal. (as a matrix).

w/ R=.001, ω_d is close to OL system, but ζ (damping ratio) is much larger...



w/ larger penalty, R=1, K is smaller & response of CL system is close to the OL sys. (CL: closed-loop) (OL: open-loop)

R↑ means u↓, but X errors ↑...

For R=1: u not exactly "zero", just much smaller in magnitude than when R=.001...

Pole placement, to determine K (gains).

(J is no longer minimized, in any way!)

$P_{desired} = [-20, -30, -100, -200]$

Pdes are set rather arbitrarily here...