Linear Quadratic Regulator (LQR)

명지대학교 전기공학과

Control & DSP 연구실

明知大學校 MYONGJI UNIVERSITY

CDSP Lab.

목 차

State Feedback Control Law

Control-law design

$$u = -Kx = -\begin{bmatrix} k_1 & k_2 & \cdots & k_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

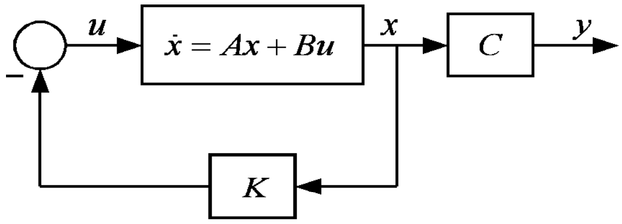
$$\dot{x} = Ax + Bu$$

$$\Rightarrow \quad \dot{x} = (A - BK)x$$

Picking the gains K so that λ (A-BK) are in desirable locations

Reference input is set to zero at this time

How to design K ?



$$G_o = C(sI - A)^{-1}B$$

$$G_q = K(sI - A)^{-1}B$$

$$G_{cl} = \frac{G(s)}{1 + G(s)}$$

Selection of Pole Locations

Dominant Second-Order Poles Method

Linear Quadratic Regulator (LQR) Design

System speed vs. Control effort

For the faster system response, the bigger control effort is necessary

How to balance between the fast response and low control effort ?

The simplified version of LQR problem

Performance index (cost function)

$$J = \int_0^{\infty} [z^2(t) + \rho u^2(t)]dt$$

ρ : weighting factor

What is the control law that minimize J for

$$\dot{x} = Ax + Bu \quad z = C_1 x$$

z : output or tracking error to be minimized

Simplified version of LQR

Linear Quadratic Regulator (LQR) Design

Choosing different values of ρ can provide us with pole locations that achieve varying balances between a fast response and a low control effort.

$$J = \int_0^\infty [z^2(t) + \rho u^2(t)]dt$$

Small ρ

➔ Low penalty on the control usage

➔ Small values of $\int_0^\infty z^2(t)dt$

➔ Faster response

Large ρ

➔ High penalty on the control usage

➔ Small values of $\int_0^\infty u^2(t)dt$

➔ Low control effort

Optimal control law is given by linear state feedback $u = -Kx$

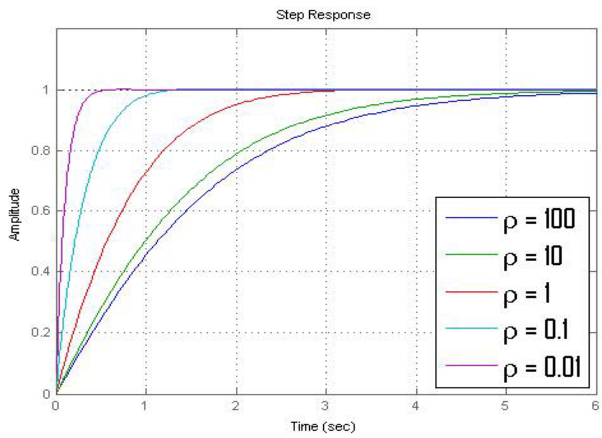
K : determined from Riccati equation

Simplified version of LQR

Simple Inverted Pendulum $A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ B = \begin{bmatrix} 0 \\ -1 \end{bmatrix},\ C = \begin{bmatrix} 2 & 1 \end{bmatrix},\ D = 0$

z = y and "ρ = 1" Case ➔ K=[−3.24 −2.73]

ρ	K	λ
100	−2.0198 −2.0124	−1.0062 + 0.0860i −1.0062 − 0.0860i
10	−2.1832 −2.1134	−1.0567 + 0.2581i −1.0567 − 0.2581i
1	−3.2361 −2.7335	−1.3668 + 0.6067i −1.3668 − 0.6067i
0.1	−7.4031 −4.9806	−2.4903 + 0.4490i −2.4903 − 0.4490i
0.01	−21.0250 −11.9185	−2.0238 −9.8947



Comparison between LQR and pole placement

LQR provides guaranteed stability margins, while the stability margins of state feedback pole placement technique are not known

Preliminaries

The sol. Of $\dot{x} = Fx$: $x = e^{Ft}x_0$

A matix is symmetric if $A^T = A$

A symmetric matrix Q is *positive (semi)definite* if

$$x^T Qx > 0 \; (x^T Qx \geq 0), \; \forall x \in R^n$$

If (A, B) is controllable (*stabilizable*), ∃ K s.t. all the eigenvalues of (A – BK) can be placed at any position (*LHP*) of the complex plane.

If (C, A) is observable (*detectable*), ∃ L s.t. all the eigenvalues of (A – LC) can be placed at any position (*LHP*) of the complex plane.

LQR Design

Linear Quadratic Regulator (LQR) problem

$$\dot{x} = Ax + Bu \qquad J = \int_0^\infty (x^T Qx + u^T Ru)dt$$
$$y = Cx$$

– Q, R : weighting matrix (Q : p.s.d. R : p.d. ➔ J ≥ 0)

– For J to achieve its min. value, both terms must go to zero.

$$J = \int_0^\infty x^T \overbrace{(Q + K^T RK)}^{S} x \, dt \qquad u = -Kx$$
$$= x_0^T \left(\int_0^\infty e^{F^T t} S \, e^{Ft} \, dt \right) x_0 \qquad F = A - BK$$
$$=: \; x_0^T P x_0, \; P \geq 0. \qquad x = e^{Ft}x_0$$

ARE for LQR Design

Controller Design : algebraic Riccati equation (ARE)

$$A^T P + PA + Q - PBR^{-1}B^T P = 0$$
$$u = -Kx, \quad K = R^{-1}B^T P$$

Matlab : K = lqr(A,B,Q,R)

When $Q = D^T D$, the closed-loop matrix F is stable if

(D, A) is detectable.

When $Q = D^T D$, the matrix P is positive definite if

(D, A) is observable.

Selection of Q and R

Bryson's rule

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

Initial choice of Diagonal matrices Q & R

Q_{ii} = 1/max. acceptable value of $[x_i^2]$

R_{ii} = 1/max. acceptable value of $[u_i^2]$

Symmetric Root Locus

$$J = \int_0^\infty (\underbrace{x^T C^T C x}_{y^T y} + u^T R u)dt$$

$$A^T P + PA + C^T C - PBR^{-1}B^T P = 0$$
$$K = R^{-1}B^T P$$

Symmetric Root Locus

$$G_o = C(sI - A)^{-1}B$$
$$G_q = K(sI - A)^{-1}B$$

$$A^T P + PA + C^T C - PBR^{-1}B^T P = 0$$

$$C^T C = (-sI - A^T)P + P(sI - A) + PBR^{-1}B^T P$$

$$K = R^{-1}B^T P$$
$$RK = B^T P,$$
$$K^T R = PB$$

$$B^T(-sI - A^T)^{-1}C^T C(sI - A)^{-1}B$$
$$= B^T P(sI - A)^{-1}B + B^T(-sI - A^T)^{-1}PB$$
$$+ B^T(-sI - A^T)^{-1}PBR^{-1}B^T P(sI - A)^{-1}B$$

$$G_o^T(-s)G_o(s) = RG_q(s) + G_q^T(-s)R + G_q^T(-s)RG_q(s)$$

$$R + G_o^T(-s)G_o(s) = R + RG_q(s) + G_q^T(-s)R + G_q^T(-s)RG_q(s)$$

$$R + G_o^T(-s)G_o(s) = \left[I + G_q^T(-s)\right]R\left[I + G_q(s)\right]$$

$$1 + \rho^{-1}G_o(-s)G_o(s) = [1 + G_q(-s)][1 + G_q(s)]$$

$$1 + \frac{1}{\rho}(-1)^{m-n}\frac{\prod^{m}(s-z_i)\prod^{m}(s+z_i)}{\prod^{n}(s-p_i)\prod^{n}(s+p_i)} = 0$$

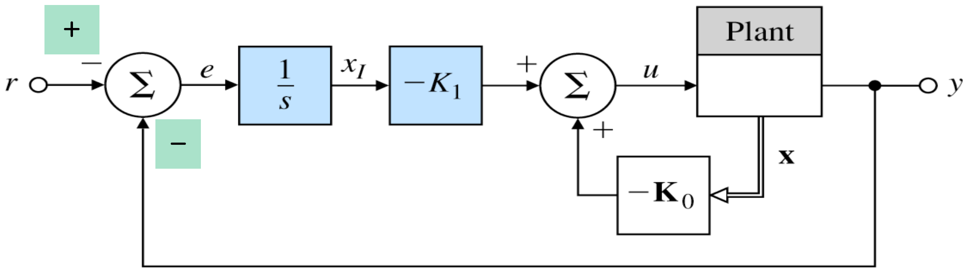
Symmetric Root Locus Ex.

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ B = \begin{bmatrix} 0 \\ -1 \end{bmatrix},\ C = \begin{bmatrix} 2 & 1 \end{bmatrix},\ D = 0$$

```
% Optimal Root Locus
clear, clc

% Simple LQR
A= [0  1; 1  0]; B= [0 ; -1]; C= [2  1]; D=0;

syso = ss(A,B,C,D);
sysos = ss(-A,B,(-1)*C, D);
syso2 = series(syso, sysos);

figure(1)
rlocus(syso2)
axis([-10 10 -1 1])
```

script | Ln 2 | Col 30

LQR + Integral Control



$$x_I \triangleq \int e\, dt = \int (r-y)dt = \int (r - Cx)dt \quad \Rightarrow \quad \dot{x}_I = r - Cx$$

$$\begin{bmatrix} \dot{x} \\ \dot{x}_I \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ x_I \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r \qquad u = -K_0 x - K_I x_I = -[K_0 \quad K_I] \begin{bmatrix} x \\ x_I \end{bmatrix}$$

$$\begin{bmatrix} \dot{x} \\ \dot{x}_I \end{bmatrix} = \begin{bmatrix} A - BK_0 & -BK_I \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ x_I \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r$$

Matlab Simulation (LQR+Integral)

$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$, $C = \begin{bmatrix} 2 & 1 \end{bmatrix}$, $D = 0$ (Simple Inverted Pendulum)

```
% Sec.2-3 / p.19 / LQR + Integral
clear, clc

A= [0  1; 1  0]; B= [0 ; -1]; C= [2  1]; D=0;
Aa = [A zeros(2,1); -C 0]; Ba = [B; 0]; Ca = [C 0; zeros(1,2) 1]; Da = [0 0]';

rho = 1;
    Q = Ca'*Ca; R = rho;
    K = lqr(Aa, Ba, Q, R)
    lambda = eig(Aa - Ba*K),
    syscl = ss(Aa-Ba*K, [0 0 1]', Ca, Da);

    figure(1)
    step(syscl), grid on,
```