Lecture 11

State Variable Feedback Design via Gain Transformation and Pole Placement

Prof. Krishna R. Pattipati

Assisted by : Kihoon Choi

Dept. of Electrical and Computer Engineering

University of Connecticut

Contact: krishna@engr.uconn.edu (860) 486-2890

ECE 6095/4121

Modeling and Digital Control of Mechatronic Systems

SVFB Design via Gain Xformation & Pole Placement

1. **“Deadbeat” controller**
2. **Continuous-Discrete Gain Tranformation**
   - Time response equivalence
   - Average gain method
   - Example-double integrator
3. **Pole Placement via SVFB Design**
   - Direct approach
   - Transformation approach
   - Ackermann formula/algorithm
   - Pole placement for MIMO systems
     - State feedback
     - output feedback (numerically not robust. So, won’t discuss)
4. **Example-Inverted Pendulum on a Cart**
   - Continuous-discrete transformation design
   - Direct digital design
5. **Implementation of High-Order Compensators**
   - State prediction
   - Comparison with Smith compensator
   - Examples
6. **Command Inputs to SVFB Systems**
   - Integral feedback

Control in State-Space

- What can be done with respect to controlling system states?

$$\underline{x}(k+1) = \Phi\,\underline{x}(k) + \Gamma u(k)$$
$$y(k) = C\underline{x}(k)$$
$$\underline{x}(0) = \text{known initial condition}$$

Equivalent discrete system matrices

$$\Phi = e^{Ah}, \qquad \Gamma = \int_0^h e^{A\sigma} d\sigma B$$

- State response

$$\underline{x}(k) = \Phi^k \underline{x}(0) + \sum_{i=0}^{k-1} \Phi^{k-1-i}\Gamma u(i)$$

- Consider k = n
  - Can we find u(0), u(1), ... , u(n–1) so that $\underline{x}(n) = \underline{\xi}$ = arbitrary vector, starting at any initial condition $\underline{x}(0)$?

$$\underline{\xi} - \Phi^n \underline{x}(0) = \sum_{i=0}^{n-1} \Phi^{n-1-i}\Gamma u(i) = \Gamma u(n-1) + \Phi\Gamma u(n-2) + \cdots + \Phi^{n-1}\Gamma u(0)$$

$$= \underbrace{\begin{bmatrix} | & | & & | \\ \Gamma & \Phi\Gamma & \cdots & \Phi^{n-1}\Gamma \\ | & | & & | \end{bmatrix}}_{H_c} \begin{bmatrix} u(n-1) \\ u(n-2) \\ \vdots \\ u(0) \end{bmatrix}$$

  - If H_c is invertible, it is possible to find the requisite {u(i)}.
  - Note: state may not necessarily stay at $\underline{\xi}$ for k > n.

Deadbeat Controller

- If system is cc the sequence {u(0), u(1), ... , u(n–1)} will drive $\underline{x}(0) \rightarrow \underline{x}(n) = \underline{\xi}$ where

$$\begin{bmatrix} u(n-1) \\ u(n-2) \\ \vdots \\ u(0) \end{bmatrix} = H_c^{-1} [\, \underline{\xi} - \Phi^n \underline{x}(0)]$$

  - Open-loop control

$$\left.\begin{aligned} u(0) &= [\,0\;\;0 \ldots 0\;\;1\,]H_c^{-1} [\, \underline{\xi} - \Phi^n \underline{x}(0)] \\ u(1) &= [\,0\;\;0 \ldots 1\;\;0\,]H_c^{-1} [\, \underline{\xi} - \Phi^n \underline{x}(0)] \\ &\vdots \\ u(n-1) &= [\,1\;\;0\;\;0 \ldots 0\,]H_c^{-1} [\, \underline{\xi} - \Phi^n \underline{x}(0)] \end{aligned}\right\} \quad \underline{x}(0),\ \underline{\text{not}}\ \underline{x}(k) \text{ is used here}$$

  - Closed-loop control via time-invariance

    "turn system on" at time "k": $\underline{x}(0) \Longleftrightarrow \underline{x}(k)$, $u(0) \Longleftrightarrow u(k)$

$$\Rightarrow\ u(k) = [\,0\;\;0 \ldots 1\,]H_c^{-1} [\, \underline{\xi} - \Phi^n \underline{x}(k)]$$

    accomplishes same control sequence but via SVFB

- Special case $\underline{\xi} = \underline{0}$

$$u(k) = -\overbrace{[\,0\;\;0 \ldots 1\,]\, H_c^{-1}\Phi^n}^{K}\, \underline{x}(k)$$

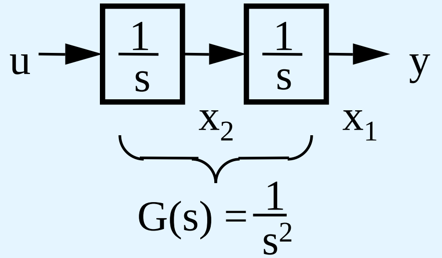
  is an SVFB control that reduces any (initial) state to $\underline{0}$ in n steps → "deadbeat controller" (unique to discrete systems)

  - CL dynamics $\quad \underline{x}(k+1) = (\Phi - \Gamma K)\, \underline{x}(k)$

$$\underline{x}(n) = (\Phi - \Gamma K)^n\, \underline{x}(0) = \underline{0}$$

  ==> $\Phi - \Gamma K$ has all eigenvalues at $z = 0$

Example - Pure Inertia Control (e.g. Satillite)



$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \underline{x}(t)$$

- Equivalent discrete system

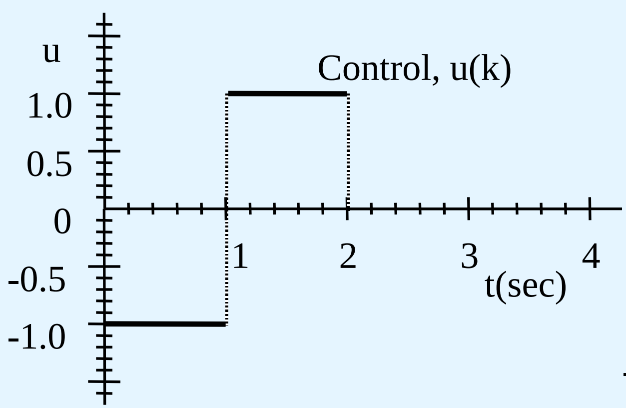
$$\underline{x}(k+1) = \overbrace{\begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix}}^{\Phi} \underline{x}(k) + \overbrace{\begin{bmatrix} h^2/2 \\ h \end{bmatrix}}^{\Gamma} u(k)$$

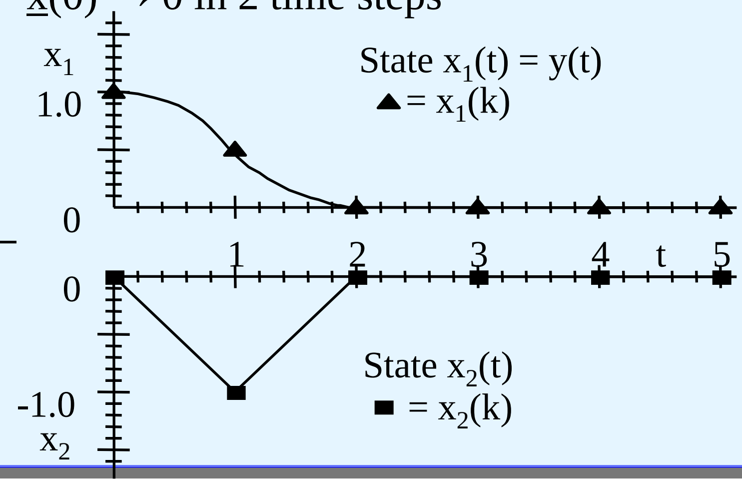
Pick h = 1

- Deadbeat controller, $u(k) = -K\,\underline{x}(k) = -K_1\, x_1(k) - K_2\, x_2(k)$

$$K = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 1/2 & 3/2 \\ 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 3/2 \end{bmatrix}$$

- Time response with $x_1(0) = 1$, $x_2(0) = 0$: $\underline{x}(0) \to 0$ in 2 time steps





Discrete SVFB Design Methods

- Continuous → discrete equivalence methods

  Given a <u>continuous</u> FB control law

  $$u(t) = K_r r(t) - K \underline{x}(t)$$

  develop from K_r, K an "equivalent" <u>discrete</u> control

  $$u(k) = \tilde{K}_r r(k) - \tilde{K} \underline{x}(k)$$

  - Idea: capitalize on earlier design for $\dot{\underline{x}} = A \underline{x} + B u$
  - Compare continuous vs. discrete time response, phase margin, closed-loop poles, etc.

- Direct digital controller design

  Find $u(k) = K_r r(k) - K \underline{x}(k)$ directly to place poles at $z = z_1, z_2, \ldots, z_n$

  (1) select $z_i = e^{s_i h}$ $\quad i = 1, 2, \ldots, n$;

  s_i = desired pole location in s-plane

  or, (2) select $z_1, z_2, \ldots, z_n$ directly

- Evaluation
  - Time response via simulation
  - Phase margin via Bode/Nyquist plot of

  $$LG(j\omega) = K(zI - \Phi)^{-1} \Gamma \Big|_{z = e^{j\omega h}}$$

  - Sensitivity to parameters, RD, etc.

Continuous →Discrete Gain Transformation Methods

C: $\dot{\underline{x}}(t) = A\,\underline{x}(t) + B\,u(t)$, $u(t) = K_r\,r(t) - K\,\underline{x}(t)$ $\longrightarrow$ ***D:*** $\underline{x}(k+1) = \Phi\,\underline{x}(k) + \Gamma\,u(k)$, $u(k) = \underline{\underline{\tilde K_r}}\,r(k) - \underline{\underline{\tilde K}}\,\underline{x}(k)$

Closed-loop

$\dot{\underline{x}}(t) = (A - BK)\,\underline{x}(t) + K_r B\,r(t)$ $\qquad$ $\underline{x}(k+1) = (\Phi - \Gamma\tilde{K})\,\underline{x}(k) + \tilde{K}_r\,\Gamma\,r(k)$

- Desire simplicity and accuracy
  - Time response of ***D*** $\overset{?}{\approx}$ time response of ***C***
  - Eigenvalues of CL ***D*** $\overset{?}{\approx}$ exp {h · eigenvalues of CL ***C*** }
- Start ***C*** at $\underline{x}(0)$ with $r(t) = r_0$. Response at $t = h$

$$\underline{x}(h) = e^{(A-BK)h}\,\underline{x}(0) + \int_0^h e^{(A-BK)\sigma} K_r B\,d\sigma \cdot r_0$$

- For discrete response

$$\underline{x}(h) = (\Phi - \Gamma\tilde{K})\,\underline{x}(0) + \tilde{K}_r\,\Gamma\,r_0$$

A) Time response equivalence:

(1) $\Phi - \Gamma\tilde{K} = e^{(A-BK)h} \;\; \rightarrow$ "solve" for $\tilde{K}$?

Only possible to obtain equivalence to $O(h^2)$

$$\tilde{K} \approx K + K(A-BK)\,h/2$$

(2) $\tilde{K}_r\Gamma = K_r \int_0^h e^{(A-BK)\sigma} B\,d\sigma$

Can only obtain equivalence to $O(h^2)$

$$\tilde{K}_r \approx \{1 - KB(h/2)\}\,K_r$$

B) Average Gain Method (Kleinman, Automatica, 1978)

- Consider $\boldsymbol{C}$ over (0, h] with $\underline{x}(0)$, and $r(t) = r_0$

$$\underline{x}(t) = e^{(A-BK)t}\,\underline{x}(0) + \int_0^t e^{(A-BK)\sigma} K_r B d\sigma \cdot r_0$$

- Control, $u_c(t)$ over (0, h], in continuous system

$$u_c(t) = K_r r_0 - K\,\underline{x}(t)$$

$$u_c(t) = \left[1 - K\int_0^t e^{(A-BK)\sigma} B d\sigma\right] K_r r_0 - K e^{(A-BK)t}\,\underline{x}(0)$$

- Discrete control over (0, h] = u(0)

$$u(0) = \tilde{K}_r r_0 - \tilde{K}\,\underline{x}(0)$$

$$\Rightarrow \text{ Pick } \tilde{K}, \tilde{K}_r \text{ so that } u(0) = \bar{u}_c = \frac{1}{h}\int_0^h u_c(t)\,dt$$

$$\bar{u}_c = \left[1 - \frac{K}{h}\int_0^h\int_0^t e^{(A-BK)\sigma} d\sigma dt B\right] K_r r_0 - \frac{K}{h}\int_0^h e^{(A-BK)t} dt\,\underline{x}(0)$$

$$\boxed{\begin{aligned}\frac{K}{h}\int_0^h\int_0^t e^{(A-BK)\sigma} d\sigma dt &= \frac{K}{h}\int_0^h \bar{\Psi}(t)dt \\ &= \frac{K}{h}\left[\int_0^h (\bar{\Phi}(t) - I)dt\right](A-BK)^{-1} \\ &= (\tilde{K} - K)(A-BK)^{-1}\end{aligned}}$$

- Discrete equivalent gains

(1) $$\tilde{K} = \frac{K}{h}\int_0^h e^{(A-BK)t} dt$$

(2) $$\tilde{K}_r = \left[1 - \frac{K}{h}\int_0^h\int_0^t e^{(A-BK)\sigma} d\sigma dt B\right] K_r = \left[1 + \left(K - \tilde{K}\right)\left(A - BK\right)^{-1} B\right] K_r$$

Computing Average Gain

- Obtain $\tilde{K}$ using c2d, then compute $\tilde{K}_r$
- Approximation for small h

$$\tilde{K} \approx K\left[I + (A-BK)\frac{h}{2} + (A-BK)^2\frac{h^2}{3!} + \cdots \right] \qquad \tilde{K}_r \approx \left\{ 1 - K\left[\frac{h}{2} + (A-BK)\frac{h^2}{3!} + \cdots \right] B \right\} K_r$$

- Average gain scheme is "good" provided

$$h \le \frac{1.0}{|\lambda_{max}(A - BK)|} \sim \frac{1.0}{\|A-BK\|}$$

Closed-loop system matrix

  - Generally requires a smaller h than does the usual criterion

$$h \le (0.5 \rightarrow 1.0)/|\lambda_{max}(A)|$$

- Using average gain $\tilde{K}$ is always better than just using K
  - but $\tilde{K}_r$ may not maintain same <u>DC gain</u> as in continuous case
- Inverse procedure: given a discrete K_d, find continuous gain K. Solve

$$K = K_d h\left[\int_0^h e^{(A-BK)t}dt \right]^{-1}$$

iteratively:

$$K_{i+1} = K_d h\left[\int_0^h e^{(A-BK_i)t}dt \right]^{-1} \text{ with } K_0 = K_d.$$

  - Generally converges in 2 - 3 iterations.
  - Useful when h is subject to change, e.g., $K_d(h_1) \rightarrow K \rightarrow K_d(h_2)$

Example – Satellite Control, $G(s)=1/s^2$

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t); \quad y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \underline{x}(t)$$

Continuous SVFB control

$$u(t) = \underset{K_r}{1.0}\, r(t) - \overbrace{[\,1 \quad 1\,]}^{K}\, \underline{x}(t)$$

Gives CL system

$$\underline{\dot{x}}(t) = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} \underline{x}(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r(t); \quad y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \underline{x}(t)$$

CL transfer function

$$\frac{y(s)}{r(s)} = \frac{1}{s^2 + s + 1}$$

CL poles at $s = -\frac{1}{2} \pm j\frac{\sqrt{3}}{2}$
($\zeta = 0.5$)

- Equivalent discrete gains, $\tilde{K}$, with $h = 0.5$

OK on "safety" requirement $h \le \dfrac{1.0}{|\lambda_{max}(A - BK)|} = \dfrac{1.0}{\sqrt{1}} = 1.0$

$$\tilde{K} = \frac{K}{h}\int_0^h e^{(A-BK)t}dt = [0.755 \quad 0.964] \;;\; \tilde{K}_r = \left[1 + \left(K - \tilde{K}\right)\left(A - BK\right)^{-1} B\right] K_r = 0.755$$

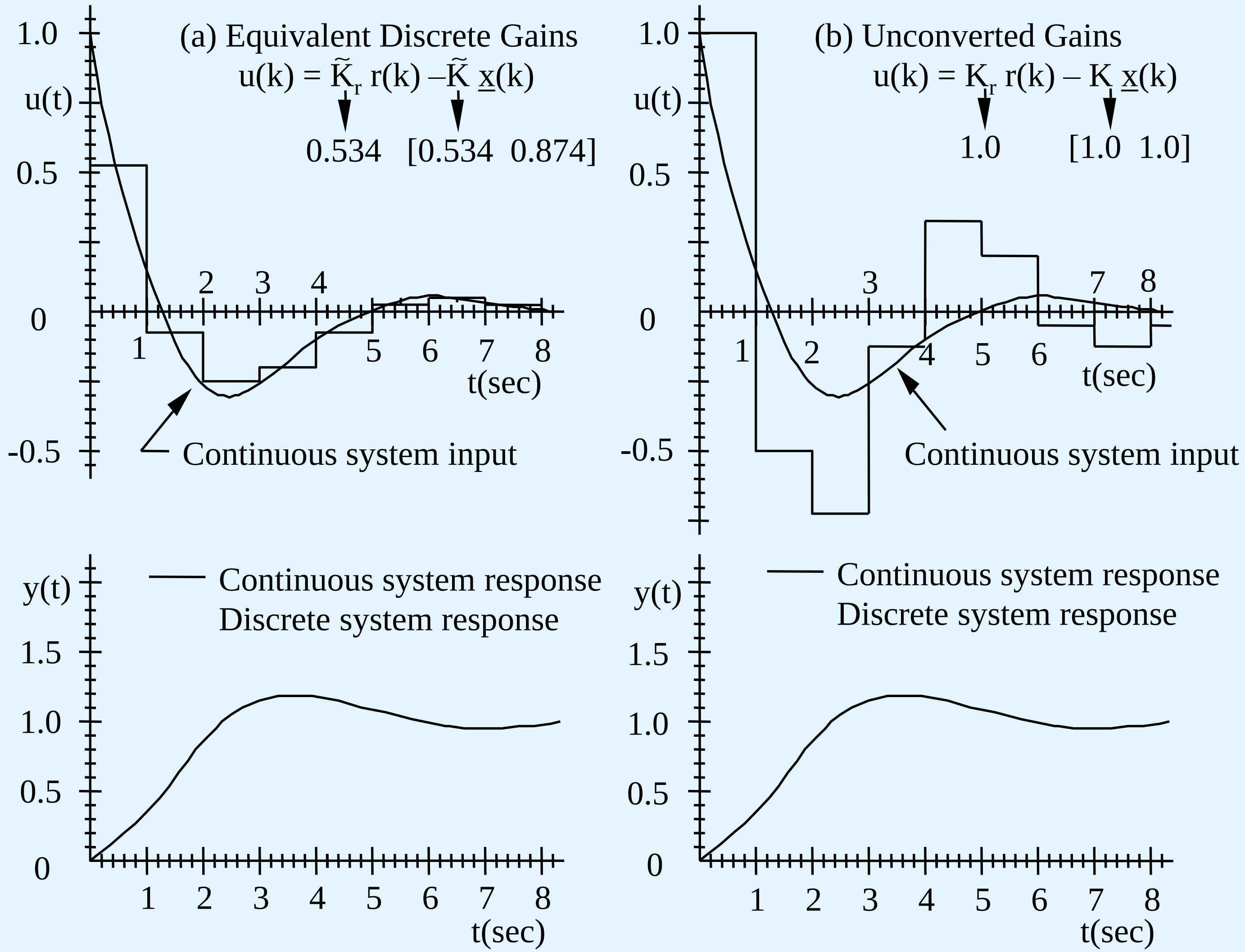
- Examine CL discrete eigenvalues (poles) of $\Phi - \Gamma\tilde{K}$
  - "Expect" poles at $z_i = e^{s_i h} = e^{-0.25 \pm j0.433} = 0.707 \pm j0.327$

$$\Phi - \Gamma\tilde{K} = \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0.125 \\ 0.5 \end{bmatrix} \begin{bmatrix} 0.755 & 0.964 \end{bmatrix} = \begin{bmatrix} 0.906 & 0.379 \\ -0.378 & 0.518 \end{bmatrix}$$

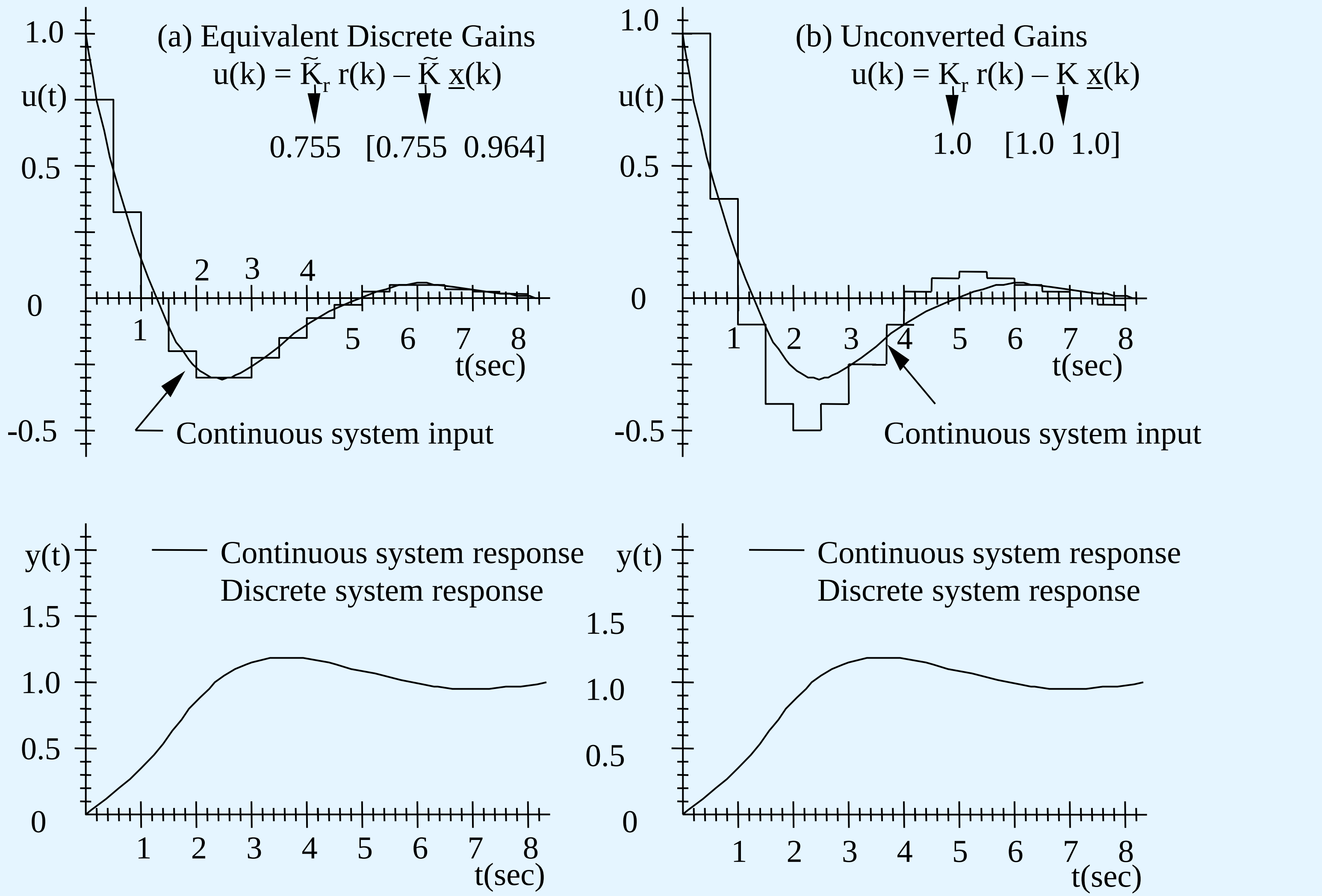
eigenvalues at $\lambda_i = 0.712 \pm j0.325$ ($\zeta \approx 0.5$)

- Without gain equivalence $\lambda_i(\Phi - \Gamma K) = 0.688 \pm j0.39$ ($\zeta = 0.41$)

Satellite Control Closed-Loop Simulation
x(0)=0, r(t)=1, h=1.0, Average Gain Equivalence

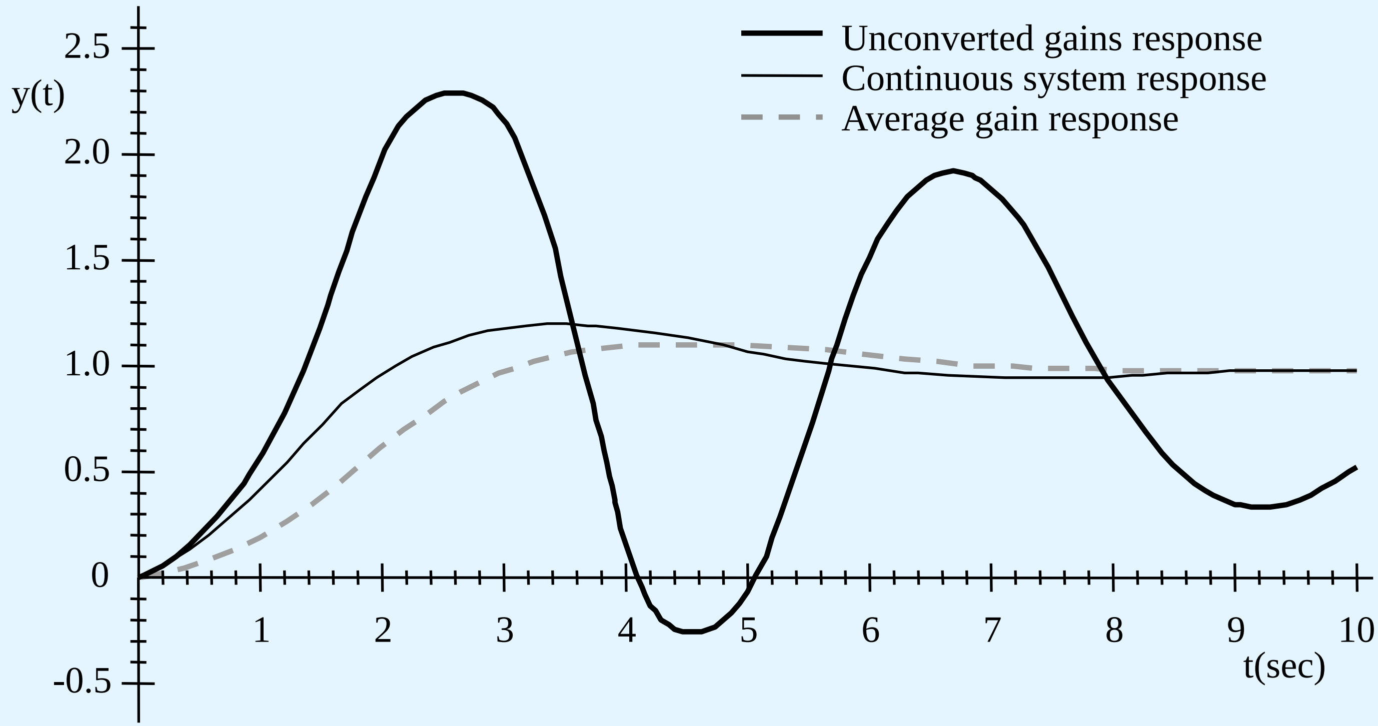


Satellite Control Closed-Loop Simulation
$\underline{x}(0)=\underline{0}$, $r(t)=1$, $\underline{h=0.5}$, Average Gain Equivalence



Satellite Control CL Simulation
$\underline{x}(0)=[0\ 0]'$, r(t)=1, h=1.8

- Even when $h > \dfrac{1.0}{|\lambda_{max}(A - BK)|}$ the equivalent (average) gains will often give a "reasonable" CL system.
- With h = 1.8, $\tilde{K} = [\ 0.261 \quad 0.683\ ]$, $\tilde{K}_r = 0.261$



- Unconverted discrete system (K = [1 1]) becomes unstable as h increase.
- CL system with average gains still hanging in, with noticeable slow-down in step response.

Summary of Equivalent Gain Method

- Average gain is best method to convert
$$K \rightarrow \tilde{K} \;;\quad K_r \rightarrow \tilde{K}_r$$
- If h ~ small ($< \dfrac{1.0}{|\lambda_{max}(A - BK)|}$) use $\tilde{K}$, $\tilde{K}_r$
- Do not simply use $\tilde{K} = K$, $\tilde{K}_r = K_r$ (instability as h increases).
- Useful if need to change h on-line frequently
  - Store K, K_r from continuous design
  - Use series approximation to obtain $\tilde{K}$, $\tilde{K}_r$ for current value of h
- Generally
  - $\tilde{K}_i$ will be smaller in magnitude than K_i.
  - Gains $\tilde{K}$ will yield discrete CL poles with a slightly smaller ω_n than original continuous system (i.e., slower response).
  - Eigenvalues of $\Phi - \Gamma\tilde{K} \approx \exp(h \cdot \text{eigenvalues of } A - BK)$.
  - Phase margin of discrete system with average gain ≈ phase margin of a discrete system with poles placed at $\exp[h \cdot \lambda_i(A - BK)]$.
  - DC gain ($r \rightarrow y$) of equivalent system not always same as ***C***.

==> may wish to pick $\tilde{K}_r$ so that DC gain of discrete CL system = DC gain of original continuous CL design, i.e., so that
$$\tilde{K}_r C [I - \Phi + \Gamma \tilde{K}]^{-1}\Gamma = -K_r C[A - BK]^{-1}B$$

- If h ≠ small, design K, K_r directly for discrete system.

State Variable Feedback Control – Direct Pole Placement (SISO)

- Discrete system design
  - Given $\underline{x}(k+1) = \Phi\, \underline{x}(k) + \Gamma\, u(k)$

    $y(k) = C\, \underline{x}(k)$

    with $|zI - \Phi| = p(z) = z^n + a_1 z^{n-1} + \cdots + a_n$ and linear feedback control structure

    $u(k) = K_r\, r(k) - K\, \underline{x}(k)$
  - Closed-loop dynamics

$$\underline{x}(k+1) = \underbrace{(\Phi - \Gamma K)}_{\bar{\Phi}\,=\,\text{discrete closed-loop matrix}}\, \underline{x}(k) + K_r\, \Gamma\, r(k)$$

$$y(k) = C\, \underline{x}(k)$$

  - Pole placement

    (1) Find FB gains $K = [K_1, K_2, \ldots, K_n]$ so that the closed-loop system matrix $\bar{\Phi}$ has eigenvalues (poles) at pre-selected locations $z_1, z_2, \ldots, z_n$*

    $\Rightarrow\ |\, zI - (\Phi - \Gamma K)\,| = (z - z_1) \cdots (z - z_n) = p_d(z)$

    $p_d(z) = z^n + d_1 z^{n-1} + \cdots + d_n$

    desired CL characteristic polynomial

    2. Adjust K_r to have unity (or some desired) DC gain from r to y

$$\frac{y(z)}{r(z)} = K_r \left[C\left(zI - \bar{\Phi}\right)^{-1} \Gamma \right]\Big|_{z=1} = 1$$

*Usually $z_i = e^{s_i h}$ where the $\{s_i\}$ are desired pole locations in s-plane.

SISO Pole Placement Methods

- Same scheme should work for either continuous or discrete problems, Φ <=> A, Γ <=> B

$$\Phi - \Gamma K \iff A - BK$$

- Direct approach
  - Expand $|zI - \Phi + \Gamma K| = z^n + f_1(K)z^{n-1} + \cdots + f_n(K)$

    [each f_i will be linear in $K_1, K_2, \ldots, K_n$]
  - Expand $p_d(z) = (z - z_1)(z - z_2)\cdots(z - z_n) = z^n + d_1 z^{n-1} + \ldots + d_n$
  - Equate coefficients and solve n linear equations, n unknowns

$$f_i(K) = d_i;\ \ i = 1, 2, \ldots, n$$

  - Useful in simple problems, some structured ones

- Example: $z_1 = 0.5 + j0.3$, $z_2 = 0.5 - j0.3$

$$\Phi = \begin{bmatrix} 1.0 & 0.2 \\ 0.2 & 1.0 \end{bmatrix} \qquad \Gamma = \begin{bmatrix} 1.0 \\ 0.5 \end{bmatrix} \qquad K = \begin{bmatrix} K_1 & K_2 \end{bmatrix}$$

$$\bar{\Phi} = \Phi - \Gamma K = \begin{bmatrix} 1.0-K_1 & 0.2-K_2 \\ 0.2-0.5K_1 & 1.0-0.5K_2 \end{bmatrix} \qquad p_d = (z - z_1)(z - z_2) = z^2 - 1.0z + 0.34$$

$$|zI - \bar{\Phi}| = z^2 + \underbrace{(-2 + K_1 + 0.5K_2)}_{f_1}z + \underbrace{(0.96 - 0.3K_2 - 0.9K_1)}_{f_2}$$

$$\left.\begin{aligned} -2 + K_1 + 0.5K_2 &= -1 \\ 0.96 - 0.9K_1 - 0.3K_2 &= 0.34 \end{aligned}\right\} \Rightarrow K_1 = 0.067,\quad K_2 = 1.867$$

  - Select K_r (e.g. so that DC gain = 1)

Transformation Approach for Pole Placement

- Let $\underline{v}(k) = T^{-1}\underline{x}(k)$ where T transforms Φ, Γ to SCF

$$\underline{x}(k+1) = \Phi\,\underline{x}(k) - \Gamma K\,\underline{x}(k) \;\Rightarrow\; \underline{v}(k+1) = \underbrace{T^{-1}\Phi T}_{\begin{bmatrix} -a_1 & -a_2 & \dots & -a_n \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \ddots & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \dots & 1 & 0 \end{bmatrix}}\underline{v}(k) - \underbrace{T^{-1}\Gamma}_{\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}} K\,\overbrace{T\,\underline{v}(k)}^{\underline{x}(k)}$$

a_i = coefficients of open-loop characteristic polynomial

- If $KT = [-a_1 + d_1, -a_2 + d_2, \dots, -a_n + d_n]$,

$$\text{then}\quad T^{-1}\Phi T - T^{-1}\Gamma K T = T^{-1}(\Phi - \Gamma K)T = \begin{bmatrix} -d_1 & -d_2 & \dots & -d_n \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \ddots & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \dots & 1 & 0 \end{bmatrix}$$

==> $\Phi - \Gamma K$ has desired characteristic polynomial $p_d(z)$ with

$$K = [-a_1 + d_1, -a_2 + d_2, \dots, -a_n + d_n]\, T^{-1}$$

- Best to solve

$$T' \begin{bmatrix} K_1 \\ K_2 \\ \vdots \\ K_n \end{bmatrix} = \begin{bmatrix} -a_1 + d_1 \\ -a_2 + d_2 \\ \vdots \\ -a_n + d_n \end{bmatrix}$$

Algorithm for Obtaining T

- Useful in general, not just for pole-placement problems

$$T = \begin{bmatrix} | & | & & | \\ \underline{t}_1 & \underline{t}_2 & \cdots & \underline{t}_n \\ | & | & & | \end{bmatrix}$$

- Generate T columnwise

$$\begin{aligned} \underline{t}_1 &= \Gamma \\ \underline{t}_2 &= \Phi\, \underline{t}_1 + a_1\Gamma \\ \underline{t}_3 &= \Phi\, \underline{t}_2 + a_2\Gamma \\ &\vdots \\ \underline{t}_n &= \Phi\, \underline{t}_{n-1} + a_{n-1}\Gamma \quad (\text{check: does } \Phi\, \underline{t}_n = -a_n\Gamma\ ?) \end{aligned}$$

- Requires computation of $\{a_i\}$ -- possible numerical problems
- T^{-1} will exist if system is completely controllable

  column $\underline{t}_k$ is a linear combination of $\Gamma, \Phi\Gamma, \dots, \Phi^{k-1}\Gamma$
- If $y(k) = C\,\underline{x}(k)$

$$y(k) = CT\,\underline{v}(k) = [\, b_1, b_2, \dots, b_n \,]\, \underline{v}(k) \quad \text{with} \quad b_i = C\,\underline{t}_i$$

- Closed-loop transfer function with $u(k) = K_r\, r(k) - K\,\underline{x}(k)$

$$\frac{y(z)}{r(z)} = K_r \frac{b_1 z^{n-1} + \cdots + b_n}{z^n + d_1 z^{n-1} + \cdots + d_n} \;;\quad b_i = C\,\underline{t}_i$$

  - for unity DC gain $= K_r = \left(1 + \sum_{i=1}^{n} d_i\right) \Big/ \sum_{i=1}^{n} b_i$
- Loop gain:

$$K(zI - \Phi)^{-1}\Gamma = \frac{\gamma_1 z^{n-1} + \cdots + \gamma_n}{z^n + a_1 z^{n-1} + \cdots + a_n} \;;\quad \gamma_i = K\,\underline{t}_i = d_i - a_i$$

Ackermann Formula

- Circumvents requirement to compute a_i

  $p_d(z) = (z - z_1)(z - z_2)\cdots(z - z_n) = z^n + d_1 z^{n-1} + \cdots + d_n$ = desired CL characteristic polynomial

  $K = [\,0 \;\; 0 \;\; \ldots \;\; 1\,]\, H_c^{-1}\, p_d(\Phi)$

  $$H_c = \begin{bmatrix} | & | & & | \\ \Gamma & \Phi\Gamma & \cdots & \Phi^{n-1}\Gamma \\ | & | & & | \end{bmatrix} = \text{Controllability matrix}$$

  $p_d(\Phi) = (\Phi - z_1 I)(\Phi - z_2 I)\cdots(\Phi - z_n I)$

  (z_i = desired poles, must be in complex conjugate pairs)

- Algorithm

  1. Set up H_c matrix one column at a time. Transpose $H_c \rightarrow H_c'$.

     Numerically unstable

  2. Solve

     $$H_c'\,\underline{q} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \quad \text{for n-vector } \underline{q}$$

     - Use any available routine for solving $A\,\underline{x} = \underline{b}$.
     - If solution fails, stop. System is not completely controllable.

  3. Evaluate $p_d(\Phi) = X$.
  4. Obtain gains $K = \underline{q}'\,X = [\,K_1, K_2, \ldots, K_n\,]$
  5. Compute K_r if needed.

Algorithm to Obtain $P_d(\Phi)$

- Compute using complex conjugate pairs to avoid complex matrices
  - e.g., if $z_3 = a + bj$, $z_4 = a - bj$

$$(z - z_3)(z - z_4) = z^2 - 2az + (a^2+b^2)$$

$$(\Phi - z_3 I)(\Phi - z_4 I) = \Phi^2 - 2a\Phi + (a^2+b^2)I$$

  - Incorporate into iterative scheme

1. initialize $X = I$, $k = 1$
2. read real & imaj. parts of roots $RA(i)$, $RB(i)$, $i = 1, 2, \ldots, n$
3. if $RB(k) = 0$: $X = X * [\Phi - RA(k)I]$
   $k = k + 1$
4. if $RB(k) \neq 0$: $X = X * [\Phi^2 - 2RA(k)\Phi + (RA^2(k) + RB^2(k))I]$
   $k = k + 2$
5. if $k \leq n$ go to 3 ; if $k = n + 1$ done

- Develop as a Subroutine GAINS
  - Can be used for continuous or discrete models
- Generally pick z_i via $e^{s_i h}$
- No restriction on h other than usual $\dfrac{(0.5 \rightarrow 1.0)}{|\lambda_{max}(A)|}$
- Deadbeat response: all $z_i = 0$ => $p_d(z) = z^n$; also $p_d(\Phi) = \Phi^n$

$$K = [\,0 \;\; 0 \;\; \cdots \;\; 1\,]\, H_c^{-1}\, \Phi^n$$

  - deadbeat gains $K_i \rightarrow \infty$ as $h \rightarrow 0$

Example – Satellite Control/Pointing

$$\underline{x}(k+1) = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} \underline{x}(k) + \begin{bmatrix} h^2/2 \\ h \end{bmatrix} u(k)$$

- Desired CL characteristic polynomial:

$$z^2 + d_1 z + d_2 = p_d(z)$$

  - if $s_i = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$: $d_1 = -2e^{-\zeta\omega_n h}\cos\left(\omega_n h\sqrt{1-\zeta^2}\right)$; $d_2 = e^{-2\zeta\omega_n h}$

- Use Ackermann algorithm: $K = [\,0 \;\; 1\,]\, H_c^{-1}\, p_d(\Phi)$

$$H_c = \begin{bmatrix} | & | \\ \Gamma & \Phi\Gamma \\ | & | \end{bmatrix} = \begin{bmatrix} h^2/2 & 3h^2/2 \\ h & h \end{bmatrix};\ \text{solve}\ \underbrace{\begin{bmatrix} h^2/2 & h \\ 3h^2/2 & h \end{bmatrix}}_{H_c'} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \Rightarrow \underline{q} = \begin{bmatrix} 1/h^2 \\ -0.5/h \end{bmatrix}$$

$$p_d(\Phi) = \begin{bmatrix} 1+d_1+d_2 & 2h+d_1 h \\ 0 & 1+d_1+d_2 \end{bmatrix} = \Phi^2 + d_1\Phi + d_2 I$$

$$K = \underline{q}' p_d(\Phi) = \begin{bmatrix} \dfrac{1+d_1+d_2}{h^2} & \dfrac{3+d_1-d_2}{2h} \end{bmatrix}$$

- Deadbeat controller: $d_1 = d_2 = 0$ ===> $K = [\,1/h^2 \quad 3/2h\,]$
  - as $h \to 0$, require excessive control energy
  - good control scheme when h is large
- For $\zeta = 0.707$, $\omega_n = 0.707$, $h = 0.5$

$$s_i = -0.5 \pm j0.5 \;\rightarrow\; z_i = e^{s_i h} = 0.53 \pm j0.29$$

$$d_1 = -1.5092,\; d_2 = 0.6065 \;\rightarrow\; K = [\,0.3894 \;\; 0.8843\,]$$

RQ Implementation

- Transform (Φ,Γ) to controller Hessenberg form, $H_p = Q^T \Phi Q$; $\Gamma_p = Q^T \Gamma = \beta \underline{e}_1$
- Perform RQ factorization of Hessenberg as follows

Set $H_1 = H_p$

For i=1,2,..,n Do

$$R_i Q_i = H_i - \lambda_i I$$

$$H_{i+1} = Q_i R_i + \lambda_i I$$

End

$$Note\; H_{i+1} = Q_i H_i Q_i^T \Rightarrow H_{n+1} = Q_n Q_{n-1} \ldots Q_2 Q_1 H_p Q_1^T Q_2^T \ldots Q_{n-1}^T Q_n^T$$

$$\Rightarrow P_d(H_p) = \prod_{i=1}^{n} (H_p - \lambda_i I) = R_1 R_2 \ldots R_n Q_n Q_{n-1} \ldots Q_2 Q_1$$

$$\Rightarrow P_d(\Phi) = Q P_d(H_p) Q^T$$

Controllability matrix: $H_c = Q\begin{bmatrix}\Gamma_p & H_p\Gamma_p & . & H_p^{n-1}\Gamma_p\end{bmatrix}$

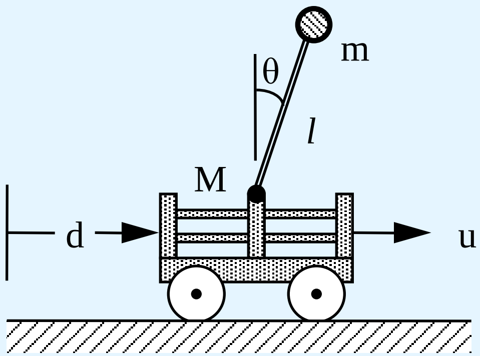
$$K = \underline{e}_n^T H_c^{-1} P_d(\Phi) = \underline{e}_n^T \underbrace{\begin{bmatrix}\Gamma_p & H_p\Gamma_p & . & H_p^{n-1}\Gamma_p\end{bmatrix}^{-1}}_{upper\ \Delta} R_1 R_2 \ldots R_n Q_n Q_{n-1} \ldots Q_2 Q_1 Q^T$$

$$= \underbrace{\alpha' \prod_{i=1}^{n} [R_i]_{nn}}_{\alpha} \underline{e}_n^T Q_n Q_{n-1} \ldots Q_2 Q_1 Q^T;\; \alpha' = 1 / \prod_{i=1}^{n-1} [H_p]_{i+1,i}$$

- Algorithm
  1. Set $H_1 = H_p$, $\alpha = \left(\prod_{i=1}^{n-1} [H_p]_{i+1,i}\right)^{-1}$
  2. For i =1,2,…,n DO
     $R_i Q_i = H_i - \lambda_i I$
     $H_{i+1} = Q_i R_i + \lambda_i I$
     $\alpha \to \alpha [R_i]_{nn}$
     End
  3. Obtain gains
     $K = \alpha \underline{e}_n^T Q_n Q_{n-1} \ldots Q_2 Q_1 Q^T$

Numerically stable
Implementation of 'place' command

Example – Inverted Pendulum on a Cart



Small angle equations:

$$\ddot{\theta}(t) = \frac{(m+M)g}{Ml}\theta(t) - \frac{u(t)}{Ml}$$

$$\ddot{d}(t) = -g\frac{m}{M}\theta(t) + \frac{u(t)}{M}$$

State equation $\underline{x} = [\theta, q, d, v]'$; $q = \dot{\theta}$, $v = \dot{d}$

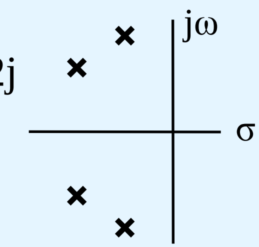
Let m = 0.1, M = 1.0, l = 1m, g ≈ 10m/sec^2

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{(m+M)g}{Ml} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -g\frac{m}{M} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ q \\ d \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{-1}{Ml} \\ 0 \\ \frac{1}{M} \end{bmatrix} u \quad \Rightarrow \quad \dot{\underline{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 11 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \end{bmatrix} \underline{x} + \begin{bmatrix} 0 \\ -1 \\ 0 \\ 1 \end{bmatrix} u$$

Open-loop eigenvalues at
$s = 0, 0, \sqrt{11}, \ -\sqrt{11}$

- Objective - Find SVFB $u = -K\underline{x}$ such that any $\underline{x}(0) \rightarrow \underline{0}$ (regulator design, r = 0)
- Continuous time design: $u(t) = -[\ 77.9\ \ -23.0\ \ -16.9\ \ -13.0\ ]\ \underline{x}(t)$ gives a stable CL system $\dot{\underline{x}} = (A - BK)\ \underline{x}$ with CL eigenvalues

$$\lambda_i(A - BK) = -2 \pm 3j, \ -3 \pm 2j$$



$$G(s)G(-s) = \frac{1}{1+(-1)^n\left(\frac{s}{\omega_c}\right)^{2n}} = \frac{1}{1+\left(\frac{-s^2}{\omega_c^2}\right)^n}$$

$$Roots: \left(\frac{-s^2}{\omega_c^2}\right) = (-1)^{1/n} \Rightarrow \left(\frac{-s^2}{\omega_c^2}\right) = e^{\frac{j(2k-1)\pi}{n}}$$

$$s_k = \omega_c e^{\frac{j(2k-1+n)\pi}{2n}}; k = 1, 2, .., n$$

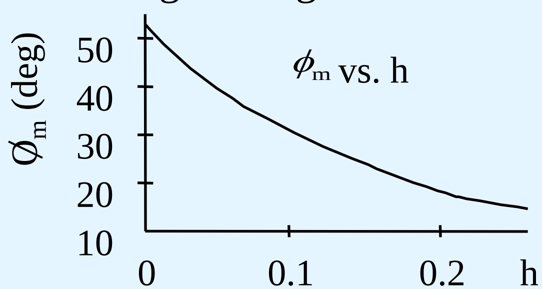
- Examine discrete time design(s)
  - equivalent (average) gains
  - direct design with $z_i = e^{s_i h}$

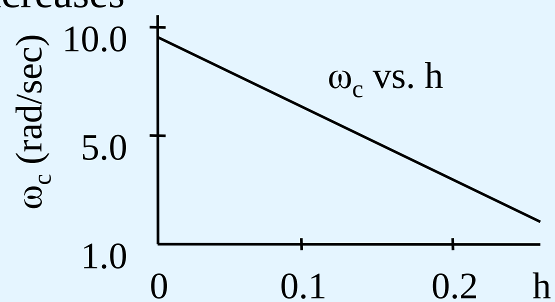
Equivalent Discrete Design, $u(k) = -\tilde{K}\underline{x}(k)$

- Expect good performance for $h \le \dfrac{1.0}{|\lambda_{max}(A - BK)|} = \dfrac{1}{\sqrt{13}} = 0.28$
- System parameters vs. h (sec)

h	$\tilde{K}_1$	$\tilde{K}_2$	$\tilde{K}_3$	$\tilde{K}_4$	ω_c	ϕ_m
0.00	-77.9	-23.0	-16.9	-13.0	9.4	53.6°
0.02	-72.6	-21.5	-15.3	-11.9	9.1	46.7
0.05	-65.1	-19.4	-13.0	-10.3	8.6	39.8
0.10	-53.5	-16.0	-9.53	-7.92	7.4	30.0
0.15	-43.2	-13.0	-6.61	-5.81	6.5	23.1
0.20	-34.1	-10.4	-4.17	-3.98	5.4	17.7
0.25	-26.3	-8.05	-2.16	-2.42	4.6	14.8

  - Large decrease in gain magnitudes as h increases





- Stability analysis
  - Discrete system has a delay of ~ h/2 sec => Reduces ϕ_m by $\omega_c h/2$
  - To avoid instability, average gain <u>lowers</u> ω_c

    => Lessens destabilizing effect of discretization delay

    => Moves ω_c into a region where $\measuredangle LG(j\omega)$ is larger
  - Examine Bode plot of LG to get ω_c and ϕ_m $\quad LG(j\omega) = \tilde{K}(zI - \Phi)^{-1}\Gamma\big|_{z=e^{j\omega h}}$

Direct Degital Design - Inverted Pendulum

- Selection of h ~ $\frac{0.5 - 1}{|\lambda_{max}(A)|} = \frac{.5-1}{\sqrt{11}}$ = 0.15 to 0.3 sec

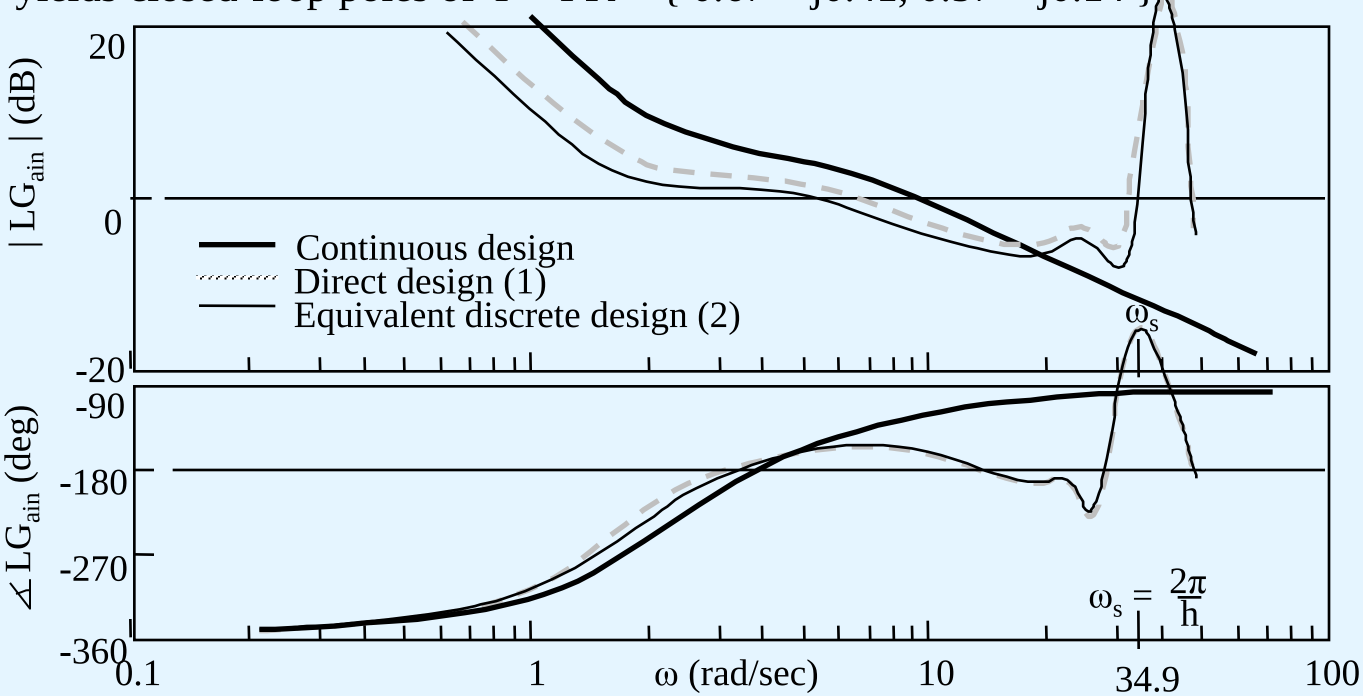
  Pick h = 0.18 sec

  (1) Pole placement @ $z_i = e^{s_i h}$ = { 0.60 ± j0.36, 0.55 ± j0.21}

  => discrete gains K = [–43.8 –13.2 –6.67 –5.91]

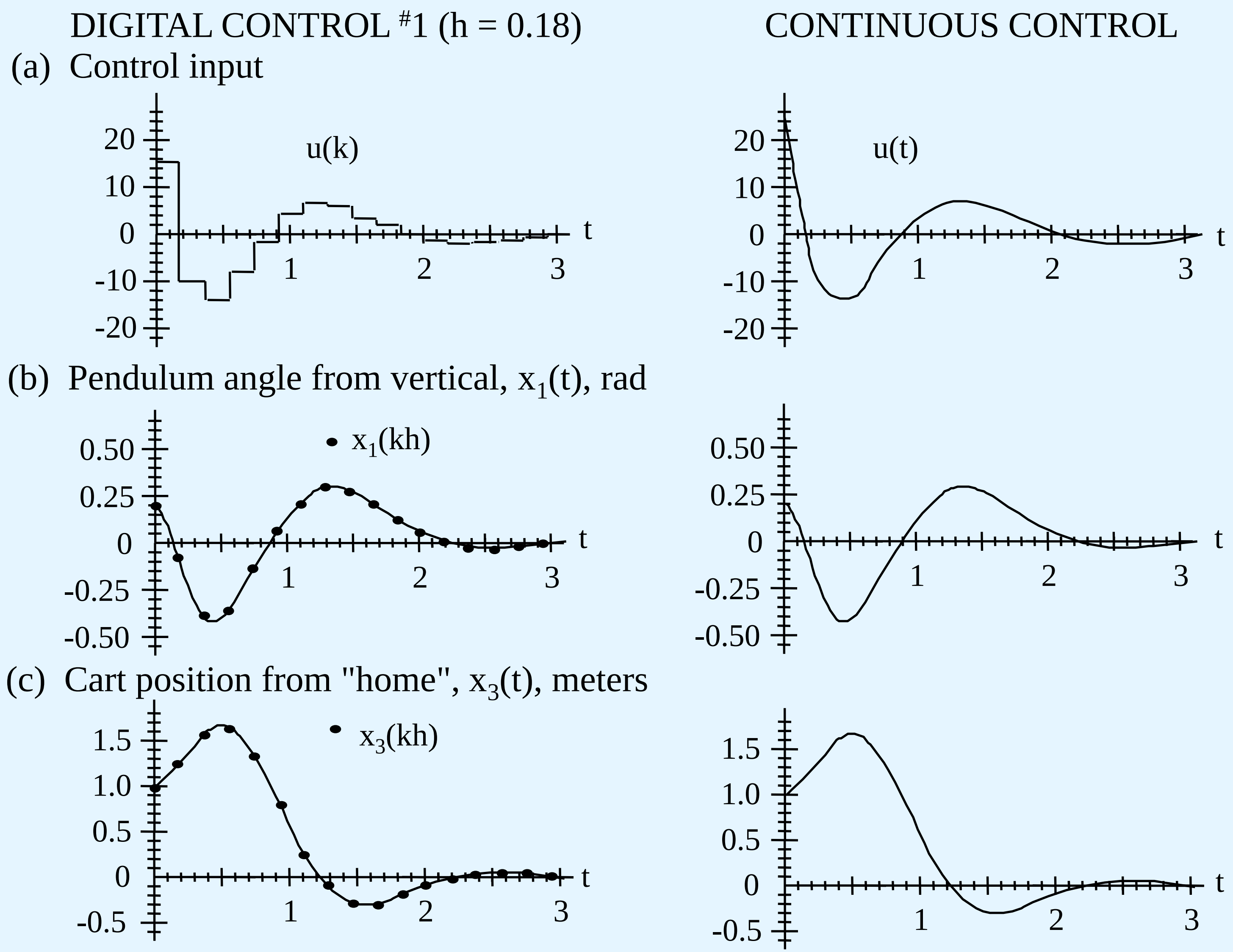
  (2) Equivalent discrete design: $\tilde{K}$ = [–37.6 –11.4 –5.09 –4.68]

  yields closed-loop poles of $\Phi - \Gamma\tilde{K}$ = { 0.67 ± j0.41, 0.57 ± j0.14 }



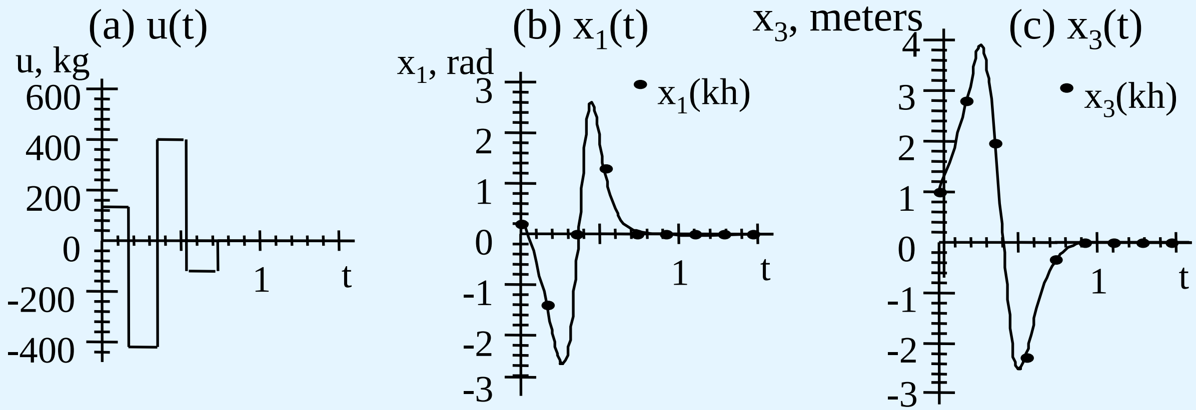
- ω_c pole placement = 6.7 vs. 5.9 for equivalent design. Both have ~ same $\phi_m \approx 19.3^\circ$! Continuous system had 56.3°.

Closed-Loop Simulaton, Inverted Pendulum, $\underline{x}(0)=[0.2, 0, 1, 0]'$

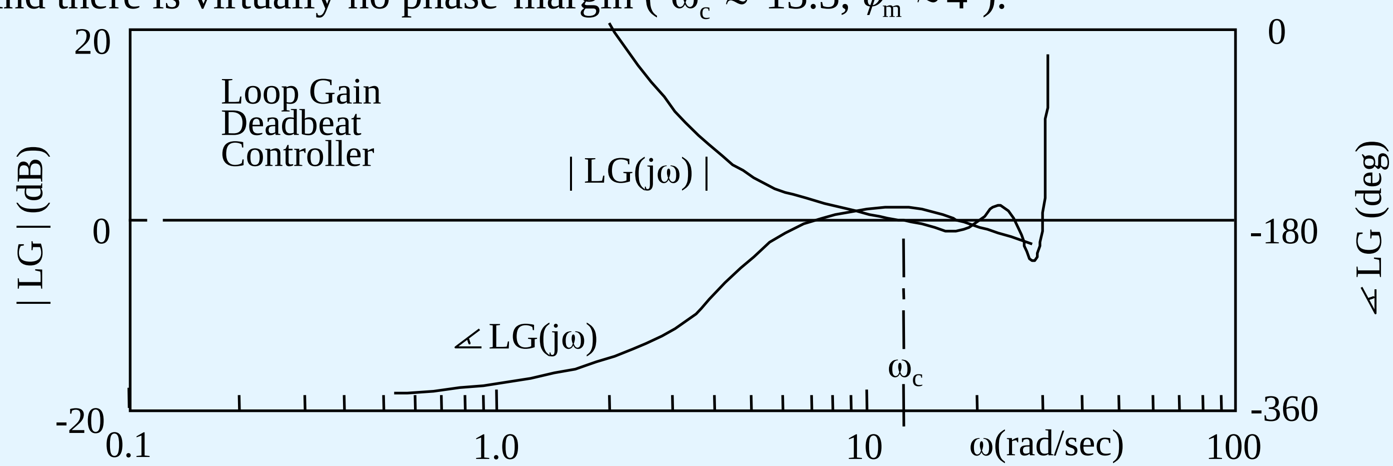


Deadbeat Controller, Inverted Pendulum, $\underline{x}(0)=[0.2, 0, 1, 0]'$

- Impossible physically in this example, but interesting... gains K = [-190.92 -53.52 -92.48 -41.62]



- Excessive control and state overshoots. Yet $\underline{x}(4) = \underline{0}$.
- And there is virtually no phase-margin ($\omega_c \approx 13.3$, $\phi_m \approx 4^o$).



- CL system has virtually no robustness properties.
- Best to reserve deadbeat for "slow" systems with h ~ large.

SVFB with Time Delay in Control, τ=Mh+ε

- First design SVFB $u(k) = -K\underline{x}(k)$ assuming $\tau = 0$.

<u>Case 1</u>: $M > 0,\ \varepsilon = 0$

$$\underline{x}(k+1) = \Phi\, \underline{x}(k) + \Gamma\, u(k-M)$$

- Predictor controller

$$u(k) = -K\underbrace{\hat{\underline{x}}(k+M)}$$ prediction of state at time $(k + M)h$ from $\underline{x}(k)$ and $u(k-1), \ldots, u(k-M)$

$$\hat{\underline{x}}(k+1) = \Phi\underline{x}(k) + \Gamma u(k-M)$$

$$\hat{\underline{x}}(k+2) = \Phi\hat{\underline{x}}(k+1) + \Gamma u(k-M+1)$$

$$\vdots \qquad = \Phi^2\underline{x}(k) + \Phi\Gamma u(k-M) + \Gamma u(k-M+1)$$

$$\hat{\underline{x}}(k+M) = \Phi^M\underline{x}(k) + \sum_{i=1}^{M}\Phi^{i-1}\Gamma u(k-i)$$

- Present control $u(k)$ will have its first effect on $\underline{x}(k+1+M)$
- Need to store past controls in a pushdown stack
- Requires a good knowledge of Φ, Γ to perform accurate propagation of $\underline{x}(k)$

<u>Case 2</u>: $M = 0,\ \varepsilon > 0\ (0 \leq \varepsilon < h)$

- Use $u(k) = -K\, \hat{\underline{x}}(kh + \varepsilon)$

$$\hat{\underline{x}}(kh+\varepsilon) = e^{A\varepsilon}\,\underline{x}(k) + \int_0^{\varepsilon} e^{A\sigma} d\sigma B u(k-1)$$

$$\Rightarrow\ u(k) = -K_x\,\underline{x}(k) - K_u\, u(k-1)$$

- Modification to structure only, propagation "hidden"
- Identical to earlier equations when $\varepsilon = h^-$ (corresp to $M = 1$)

Implementation of Delay Compensator, General Case

- Basic Idea: construct u(k) so that τ sec later, $u(kh+\tau) \approx -K\underline{x}(kh+\tau)$

  $\Rightarrow$ input $u(k) = -K\hat{\underline{x}}(kh+\tau)$ now
- Algorithm: Enter with $\underline{x}(k)$ = current state measurement

  $u(k-1)$ = last control generated
  - Need to set up a delay stack (initialized to zero)

    $$V = [v_0, v_1, \cdots, v_M] = [\,u(k-1-M) \,\vdots\, u(k-M) \,\vdots\, \cdots \,\vdots\, u(k-1)\,]$$
  - Propagate current state ahead ε sec: $\underline{x}_e = \hat{\underline{x}}(kh+\varepsilon)$

    $$\hat{\underline{x}}(kh+\varepsilon) = e^{A\varepsilon}\underline{x}(k) + \int_0^{\varepsilon} e^{A\sigma} d\sigma B u(k-1-M)$$
  - Propagat $\underline{x}_e$ ahead M time steps and apply control $u = -K\underline{x}_e$

Set $v_M = u(k-1)$, $\underline{x}_e = \underline{x}(k)$ — Put last control on stack

$\underline{x}_e \leftarrow e^{A\varepsilon}\underline{x}_e + \int_0^{\varepsilon} e^{A\sigma} d\sigma B v_0$ — State propagation (bypass if $\varepsilon = 0$)

Do for $i = 1, \ldots, M$: $\underline{x}_e \leftarrow \Phi \underline{x}_e + \Gamma v_i$ — State propagation (bypass if $M = 0$)

$\underline{x}_e = \hat{\underline{x}}(kh + \varepsilon + Mh) = \hat{\underline{x}}(kh + \tau)$

$u(k) = K_r r(k) - K\underline{x}_e$ — Control

Do for $i = 1, \ldots, M$: $v_{i-1} = v_i$ — Stack pushdown

- Algorithm can be rearranged for greater efficiency (need to store $\Phi^j\Gamma$)

Comparison with Smith Predictor Structure (ε=0)

- Define system "model"

$$\underline{\xi}(k+1) = \Phi\,\underline{\xi}(k) + \Gamma\,u(k)$$

$$\underline{\xi}(k) \sim \text{crude estimate of } \underline{x}(k + M)$$

$$\Rightarrow \underline{\xi}(k) = \Phi^M \underline{\xi}(k-M) + \sum_{i=1}^{M} \Phi^{i-1}\Gamma u(k-i)$$

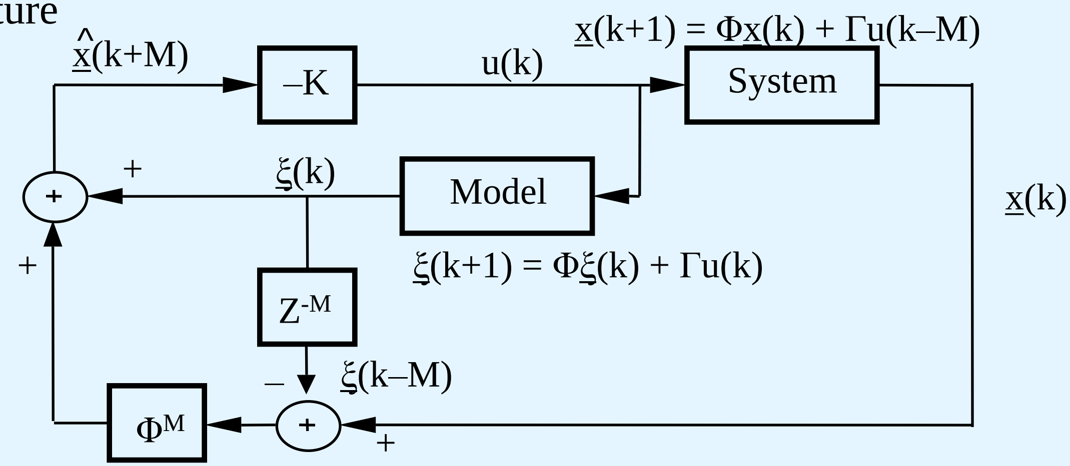
$$\Rightarrow \sum_{i=1}^{M} \Phi^{i-1}\Gamma u(k-i) = \underline{\xi}(k) - \Phi^M \underline{\xi}(k-M)$$

- $\hat{x}(k+M)$ = Prediction estimate:

$$\hat{\underline{x}}(k+M) = \Phi^M [\underline{x}(k) - \underline{\xi}(k-M)] + \underline{\xi}(k)$$

$$\text{control } u(k) = -K\,\hat{\underline{x}}(k+M)$$

- Loop structure



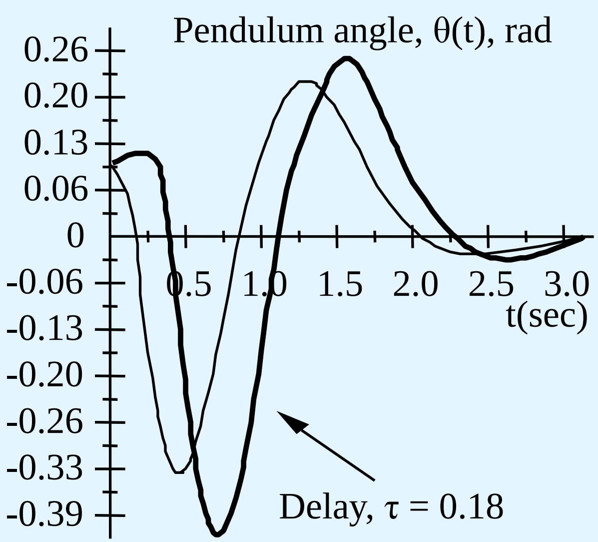
- Nearly identical to Smith predictor ($\underline{x} \sim y$)
  - Preferable to use state propagation formula, especially if system is open-loop unstable and Φ, Γ are not perfectly known.

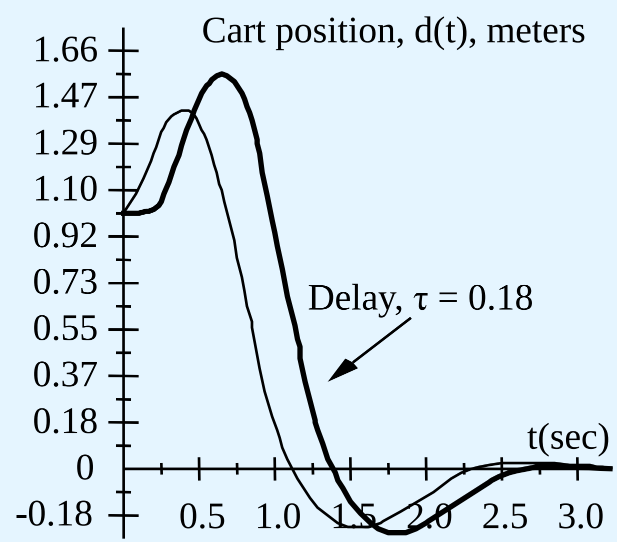
Example – Inverted Pendulum

h = 0.18 sec K = [-43.8 -13.2 -6.67 -5.91] (gains obtained via pole placement)

$\omega_c = 6.7,\ \phi_m = 19.3^o \Rightarrow \tau_{max} = \phi_m/\omega_c \approx 0.05$ sec

- Select $\tau = 0.18$ (corresponds to M = 1, $\varepsilon = 0$).
  System is highly unstable unless delay is compensated.
- Simulation $\underline{x}(0) = [\ 0.1\ \ 0\ \ 1.0\ \ 0\ ] = [\ \theta,\ \dot{\theta},\ d,\ \dot{d}]$
  compare with response of system with no delay





- System "drifts" for first τ sec, then is controlled to zero.
- In ideal case, state response for k > M is identical to an undelayed response with an initial condition $\underline{x}(M) = \Phi^M \underline{x}(0)$, and shifted by Mh sec.

  => from $k \geq M$, predictor control is "perfect" (assuming you know Φ and Γ).

Robust MIMO Pole Placement : State Feedback - 1

- Kautsky's Algorithm
- In MIMO, we have *mn* degrees of freedom, but only *n* pole locations. Use the remaining degrees of freedom to *minimize the conditioning of the closed-loop eigen vector matrix*.

Recall Eigen value conditioning:

$$s_j = \frac{d\lambda_j}{d\varepsilon} = \frac{\| \underline{y}_j \|_2 \| \underline{x}_j \|_2}{| \underline{y}_j^T \underline{x}_j |}; \varepsilon = parameter\ in\ \Phi, \Gamma, K$$

$\underline{x}_j$ = Right eigen vector of $\Phi - \Gamma K$; $\underline{y}_j$ = Left eigen vector of $\Phi - \Gamma K$

$$Metrics: (i)\ J = \min_K \max_j s_j; (ii)\ J = \min_K \left(\sum_{i=1}^{n} s_j^2 \right)^{1/2}; (iii)\ \kappa_2(T) = \| T \|_2 \| T^{-1} \|_2$$

Let $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_n)$ be pole locations

If T is the matrix of eigen vectors: $(\Phi - \Gamma K)T = T\Lambda \Rightarrow \Gamma K = \Phi - T\Lambda T^{-1}$

Suppose we do QR decomposition of $\Gamma : \Gamma = Q_1 R_1 = [Q_1 \quad Q_2] \begin{bmatrix} R_1 \\ 0 \end{bmatrix}$

$so,\ R_1 K = Q_1^T \left(\Phi - T\Lambda T^{-1} \right) \Rightarrow K = R_1^{-1} Q_1^T \left(\Phi - T\Lambda T^{-1} \right).\ Also,\ Q_2^T \Gamma K = 0 = Q_2^T \left(\Phi - T\Lambda T^{-1} \right)$

One way of picking T is find $N\left[\left(\Phi - \lambda_j I \right)^T Q_2 \right]$; $j = 1, 2, .., n$ and pick directions from these.

So, $\left(\Phi - \lambda_j I \right)^T Q_2 = \left[L_j \quad V_j \right] \begin{bmatrix} \Delta_j \\ 0 \end{bmatrix} = L_j \Delta_j \Rightarrow V_j \in N\left[\left(\Phi - \lambda_j I \right)^T Q_2 \right]$

One approach: Form an arbitray T. Replace column $\underline{t}_j \ni new\ \underline{t}_j = \dfrac{V_j V_j^T \underline{u}_j}{\| V_j^T \underline{u}_j \|_2}$ *where* $\underline{u}_j \perp R[\underline{t}_1, \underline{t}_2, .., \underline{t}_{j-1}, \underline{t}_{j+1}, .., \underline{t}_n]$

Robust Pole Placement Algorithm

- Kautsky's Algorithm
  - Step 1: Do QR decomposition of $\Gamma = Q_1 R_1 = [Q_1 \quad Q_2]\begin{bmatrix} R_1 \\ 0 \end{bmatrix}$
  - Step 2: For j=1,2,..,n Do

    Compute V_j by doing QR decomposition of

    $$\left[\left(\Phi - \lambda_j I\right)^T Q_2\right] = \left[L_j \quad V_j\right]\begin{bmatrix} \Delta_j \\ 0 \end{bmatrix}$$

    End
  - Step 3: Select from $\{V_j\}_{j=1}^n$ one from each an independent vector set to form *T*.

    Now iterate until $k_2(T)$ stabilizes.

$$[\underline{t}_1, \underline{t}_2, .., \underline{t}_{j-1}, \underline{t}_{j+1}, .., \underline{t}_n] = \left[U_j \quad \underline{u}_j\right]\begin{bmatrix} Z_j \\ 0 \end{bmatrix} \Rightarrow new\, \underline{t}_j = \frac{V_j V_j^T \underline{u}_j}{\| V_j^T \underline{u}_j \|_2}$$

$$Note: s_j = \frac{1}{|\underline{u}_j^T \underline{t}_j|} = \frac{1}{\| V_j^T \underline{u}_j \|_2}$$

MATLAB **place** implements this

- Compute gains:

$$K = R_1^{-1} Q_1^T \left(\Phi - T \Lambda T^{-1}\right)$$

Application Example

- Chemical Reactor Example: n=4, m=2

- Continuous-time system

$$A=\begin{bmatrix} 1.3800 & -0.2077 & 6.7150 & -5.6760 \\ -0.5814 & -4.2900 & 0 & 0.6750 \\ 1.0670 & 4.2730 & -6.6540 & 5.8930 \\ 0.0480 & 4.2730 & 1.3430 & -2.1040 \end{bmatrix}; B=\begin{bmatrix} 0 & 5.6790 \\ 1.1360 & 1.1360 \\ 0 & 0 \\ -3.1460 & 0 \end{bmatrix}; \lambda_i(A)=\begin{bmatrix} 1.9910 \\ 0.0635 \\ -5.0566 \\ -8.6659 \end{bmatrix}; \text{ desired } \lambda_i(\bar{A})=\begin{bmatrix} -0.200 \\ -0.500 \\ -5.0566 \\ -8.6659 \end{bmatrix}$$

- $h = 0.2/\|A\| = 0.0154 \Rightarrow$ select $h = 0.01\,\text{sec}$

- Discretized system

$$\Phi=\begin{bmatrix} 1.0142 & -0.0018 & 0.0651 & -0.0546 \\ -0.0057 & 0.9582 & -0.0001 & 0.0067 \\ 0.0103 & 0.0417 & 0.9363 & 0.0563 \\ 0.0004 & 0.0417 & 0.0129 & 0.9797 \end{bmatrix}; \Gamma=\begin{bmatrix} 0.0009 & 0.0572 \\ 0.0110 & 0.0110 \\ -0.0007 & 0.0005 \\ -0.0309 & 0.0003 \end{bmatrix}; \lambda_i(\Phi)=\begin{bmatrix} 1.0201 \\ 1.0006 \\ 0.9507 \\ 0.9170 \end{bmatrix}; \text{desired } \lambda_i(\bar{\Phi})=\begin{bmatrix} 0.9980 \\ 0.9950 \\ 0.9507 \\ 0.9170 \end{bmatrix}$$

- Gains $K=\begin{bmatrix} 0.1325 & -0.9186 & 0.1731 & -0.0692 \\ 0.5030 & 0.6180 & 0.4966 & -0.3457 \end{bmatrix}; T=\begin{bmatrix} -0.5261 & 0.0834 & 0.7273 & -0.5877 \\ 0.0428 & 0.8082 & -0.2123 & 0.2540 \\ 0.8214 & 0.2057 & 0.4402 & 0.5555 \\ -0.2162 & -0.5455 & 0.4819 & 0.5306 \end{bmatrix}; \kappa_2(T) = 2.2537$

Summary of Pole Placement Design by SVFB

- Valid for continuous or discrete design
  - Ackermann formula to find K
  - Transform $K \rightarrow \tilde{K}$ if design developed on $\boldsymbol{C}$ and h ~ small
  - Need to select all n pole locations
- SVFB does not modify system zeros
  - Can combine compensator H(z) <u>and</u> SVFB to adjust/move zeros

=> <u>Advantages</u>

- Straightforward design methodology
- Direct control over CL pole locations
- Uses all available information in the feedback
- Ability to design deadbeat control
- Possible to extend to MIMO systems, but cumbersome

=> <u>Disadvantages</u>

- Need to measure or estimate all states
- More complex design than series compensation
- No <u>direct</u> control over CL time response (still requires trial and error with CL simulation)
- Not always clear where to place all n poles
- No direct control over ϕ_m, ω_c

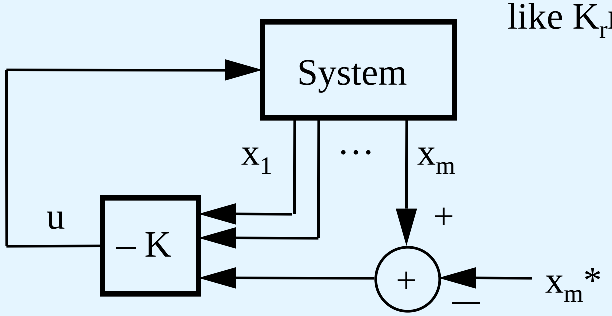
Command Inputs to SVFB Systems

- Consider continuous case for simplicity

$$\dot{\underline{x}}(t) = A\,\underline{x}(t) + Bu(t)$$
$$u(t) = -K\underline{x}(t)$$

  - Closed-loop $A-BK$ has poles at desired locations
- Desire x_m = m-th component of $\underline{x}(t) \to x_m^*$ in steady-state
- Idea: Use control

$$u(t) = -K_1x_1(t) - \cdots - K_m\,[x_m(t) - x_m^*] - \cdots - K_nx_n(t)$$
$$u(t) = \underbrace{K_m x_m^*}_{\text{like } K_r r(t) \text{ with } r(t) = \text{constant}} - K\underline{x}(t)$$



$$\dot{\underline{x}}(t) = (A - BK)\,\underline{x}(t) + BK_mx_m^*$$

- Steady-state $\underline{x}$: $\underline{x}_{ss} = -(A-BK)^{-1} B K_m x_m^*$

$$\underline{x}_{m,ss} = -\underline{e}_m'(A-BK)^{-1} B K_m x_m^* \;;\; \underline{e}_m' = [\,0\; 0 \ldots \overset{\text{m-th element}}{\overset{\downarrow}{1}} \ldots 0\,]$$

  - Generally $x_{m,ss} \neq x_m^*$

    => Adjust $x_m^* \to \beta x_m^*$ and pick β so that $x_{m,ss} = x_m^*$

  or consider use of <u>integral control</u>

$$u(t) = -Kx_1(t) - \cdots - K_m\left[x_m(t) - x_m^*\right] - \cdots - K_nx_n(t) - K_{n+1}\int_0^t\left[x_m(\sigma) - x_m^*\right]d\sigma$$

Integral Control in SVFB

- Define $x_m(t)|_{new} = x_m(t)|_{old} - x_m^* \triangleq$ error in state $x_m \Rightarrow \dot{\underline{x}}(t) = A\,\underline{x}(t) + \underline{a}_m x_m^* + Bu(t)$

  $\underline{a}_m$ = m-th column of A, (often $\underline{a}_m = \underline{0}$, especially if x_m is a position variable)

- Define $\quad x_{n+1}(t) = \int_0^t x_m(\sigma)\,d\sigma = \left\{\begin{array}{l}\text{integral of error in } x_m\\ \text{from desired ss value}\end{array}\right.$

  $\dot{x}_{n+1}(t) = x_m(t)\,; \quad x_{n+1}(0) = 0$

- Augmented (n +1)-st order system, $\underline{x}_a = [\underline{x},\ x_{n+1}]'$

$$\begin{bmatrix}\dot{\underline{x}}\\ \dot{x}_{n+1}\end{bmatrix} = \left[\begin{array}{c|c} A & 0\\ \hline \underline{e}_m' & 0\end{array}\right]\begin{bmatrix}\underline{x}\\ x_{n+1}\end{bmatrix} + \left[\begin{array}{c}\underline{a}_m\\ \hline 0\end{array}\right]x_m^* + \left[\begin{array}{c}B\\ \hline 0\end{array}\right]u$$

$$\dot{\underline{x}}_a(t) = A_a\underline{x}_a(t) + B_a u(t) + \underline{\tilde{a}}_m x_m^*$$

- Augmented system may not be controllable. Examine

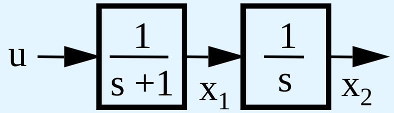
$$[\,B_a\ \ A_aB_a \cdots A_a^nB_a] = \begin{bmatrix} B & AB & \cdots & A^nB\\ 0 & \underline{e}_m'B & \cdots & \underline{e}_m'A^{n-1}B\end{bmatrix} = \begin{array}{l}(n+1) \text{ x } (n+1)\\ \text{Controllability matrix}\end{array}$$

- Selection of $u(t) = -K_a\,\underline{x}_a(t) = -K\,\underline{x}(t) - K_{n+1}x_{n+1}(t)$
  - Design K as before, to place poles of $A - BK = \bar{A}$
  - K_{n+1} ~ small gain on integral error
- CL characteristic polynomial, $|\,sI - A_a + B_a K_a\,|$

$$\begin{vmatrix} sI - \bar{A} & BK_{n+1}\\ -\underline{e}_m' & s\end{vmatrix} = |\,sI - \bar{A}\,|\cdot\underbrace{|\,s + \underline{e}_m'(sI - \bar{A})^{-1}B\,K_{n+1}\,|}_{\sim\,(s - \underline{e}_m'\bar{A}^{-1}BK_{n+1})\text{ for small } K_{n+1}} = |sI - \bar{A}_a|$$

  - Select K_{n+1} so that pole is in LHP.

Example – Integral Control



x_1 = Motor shaft velocity

x_2 = Shaft angular position

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}-1 & 0\\ 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}1\\ 0\end{bmatrix}u$$

$u = -[\,1\ \ 2\,]\,\underline{x}$ places CL poles at $s = -1 \pm j$

(1) Desire $x_1(t) \rightarrow x_1^*$ in ss (obvious problem since $x_2 \rightarrow \infty$)

Introduce integral control, $x_1 \triangleq x_{1,e}$, obtain augmented state equation

$$\underline{\dot{x}}_a = \left[\begin{array}{ccc}-1 & 0 & 0\\ 1 & 0 & 0\\ \hdashline 1 & 0 & 0\end{array}\right]\underline{x}_a + \left[\begin{array}{c}-1\\ 1\\ \hdashline 0\end{array}\right]x_1^* + \left[\begin{array}{c}1\\ 0\\ \hdashline 0\end{array}\right]u \ ;\ |H_c| = \begin{vmatrix}1 & -1 & 1\\ 0 & 1 & -1\\ 0 & 1 & -1\end{vmatrix} = 0!$$

- Uncontrollable, cannot have a stable system if x_1 = constant

(2) Desire $x_2(t) \rightarrow x_2^*$ in ss. Introduce integral control, $x_2 \rightarrow x_{2,e}$

$$\underline{\dot{x}}_a = \left[\begin{array}{ccc}-1 & 0 & 0\\ 1 & 0 & 0\\ \hdashline 0 & 1 & 0\end{array}\right]\underline{x}_a + \left[\begin{array}{c}0\\ 0\\ \hdashline 0\end{array}\right]x_2^* + \left[\begin{array}{c}1\\ 0\\ \hdashline 0\end{array}\right]u \ ;\ |H_c| \neq 0$$

$$u = -[\,K_1\ \ K_2\ \ K_3\,]\,\underline{x}_a = -K_1x_1 - K_2x_2 - K_3\int_0^t x_2(\sigma)\,d\sigma$$

let $[\,K_1\ \ K_2\,] = [\,1\ \ 2\,]$ = same as before for primary poles; $K_3 = \varepsilon$ ~ small

$$|sI - \bar{A}_a| = s(s^2 + 2s + 2) + \varepsilon \ \sim\ (s^2 + 2s + 2)(s + \varepsilon/2)$$

- Too much gain on $\int$-term is NG when simply added in

Alternate approach - Pick $[\,K_1\ \ K_2\ \ K_3\,]$ to place all 3 CL poles in LHP -- but this destroys association $K_3 \leftrightarrow$ new pole @ $s \sim 0$