STATE-FEEDBACK CONTROL

6.1: State-feedback control

- We are given a particular system having dynamics

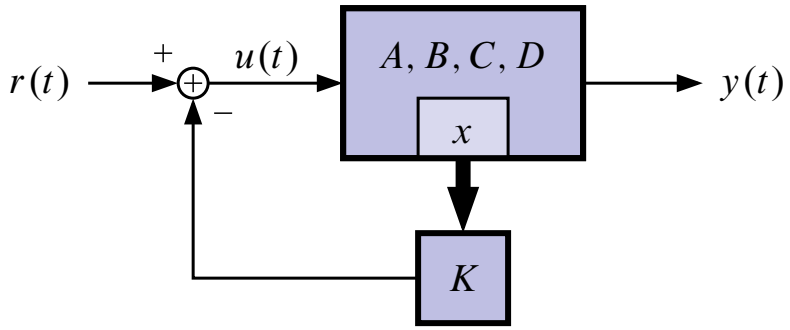
$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t).$$

- We know that open-loop system poles are given by eigenvalues of A.
- Want to use input $u(t)$ to change the dynamics.
- Will assume the form of linear state feedback with gain vector K

$$u(t) = r(t) - Kx(t), \quad K \in \mathbb{R}^{1\times n}.$$

- We assume that $x(t)$ is known exactly: then, we have



- Substitute:

$$\dot{x}(t) = Ax(t) + B(r(t) - Kx(t))$$

$$= (A - BK)x(t) + Br(t)$$

$$y(t) = Cx(t) + Du(t).$$

- For now we talk about regulation ($r(t) = 0$) and generalize later.

- For now we consider SISO systems, and generalize later.

OBJECTIVE: Design K so that $A_{CL} = A - BK$ has some nice properties. For example,

- Perhaps A is unstable. Design A_{CL} to be stable.
- Or, might want to put two poles at $-2 \pm j$. (Pole placement.)

- There are n parameters in the gain vector K and n eigenvalues of A. So, what can we achieve?

EXAMPLE: Consider the (unstable) system

$$\dot{x}(t) = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(t).$$

$$\det(sI - A) = (s-1)(s-2) - 1 = s^2 - 3s + 1.$$

- Let

$$u(t) = -\begin{bmatrix} k_1 & k_2 \end{bmatrix} x(t) = -Kx(t)$$

$$A_{CL} = A - BK = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} k_1 & k_2 \end{bmatrix}$$

$$= \begin{bmatrix} 1-k_1 & 1-k_2 \\ 1 & 2 \end{bmatrix}.$$

- So, $\det(sI - A_{CL}) = s^2 + (k_1 - 3)s + (1 - 2k_1 + k_2)$.
- By choosing k_1 and k_2, we can put $\mathrm{eig}(A_{CL})$ *anywhere* in the complex plane (in complex-conjugate pairs, that is!)
- For example, how can we place poles at -5, -6?

- Compare desired closed-loop characteristic equation

$$(s+5)(s+6) = s^2 + 11s + 30$$

with

$$\det(sI - A_{CL}) = s^2 + (k_1 - 3)s + (1 - 2k_1 + k_2).$$

- So,

$$k_1 - 3 = 11, \quad \text{or, } k_1 = 14$$

$$1 - 2k_1 + k_2 = 30, \quad \text{or, } k_2 = 57.$$

- $K = [14\ 57]$.
- So, with the n parameters in K, can we always relocate all n $\text{eig}(A_{CL})$?
  - Most physical systems, qualified yes.
  - Mathematically, EMPHATIC NO!
- Boils down to whether or not the system is *controllable.* That is, if every internal system mode can be excited by inputs, either directly or indirectly.

EXAMPLE: Consider the system

$$\dot{x}(t) = \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(t) \qquad \text{and} \qquad u(t) = -Kx(t).$$

$$A_{CL} = A - BK = \begin{bmatrix} 1 - k_1 & 1 - k_2 \\ 0 & 2 \end{bmatrix}$$

$$\det(sI - A_{CL}) = (s - 1 + k_1)(s - 2).$$

- Feedback of the state cannot move the pole at $s = 2$. System *cannot* be stabilized via state feedback. (Note that the system is already in Kalman form, and the uncontrollable mode has eigenvalue 2).

6.2: Bass–Gura pole placement

FACT: If $\{A, B\}$ is controllable, then we can arbitrarily assign the eigenvalues of A_{CL} using state feedback. More precisely, given any polynomial $s^n + \alpha_1 s^{n-1} + \cdots + \alpha_n$ there exists a (unique for SISO) $K \in \mathbb{R}^{m \times n}$ ($m =$ number of system inputs $= 1$ for SISO) such that

$$\det(sI - A + BK) = s^n + \alpha_1 s^{n-1} + \cdots + \alpha_n.$$

COROLLARY: If $\{A, B\}$ is controllable, we can find a state feedback for which the closed-loop realization is stable.

PROOF OF FACT: We prove this by solving for the state feedback vector K to put the poles where we desire.

- First, suppose $\{A, B\}$ is controllable.
- To make a hard problem easier, we then transform $\{A, B, C, D\}$ into *controller canonical form.* (Then solve problem in this form, then convert back to original form).
- That is, find T such that

$$T^{-1}AT = A_c = \begin{bmatrix} -a_1 & -a_2 & \cdots & -a_n \\ 1 & & & 0 \\ & \ddots & & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix} \quad \text{and} \quad T^{-1}B = B_c = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

  where $\det(sI - A) = s^n + a_1 s^{n-1} + \cdots + a_n$.

  - We defer calculation of T for now.

- Let's apply state-feedback K_c to the controller realization.

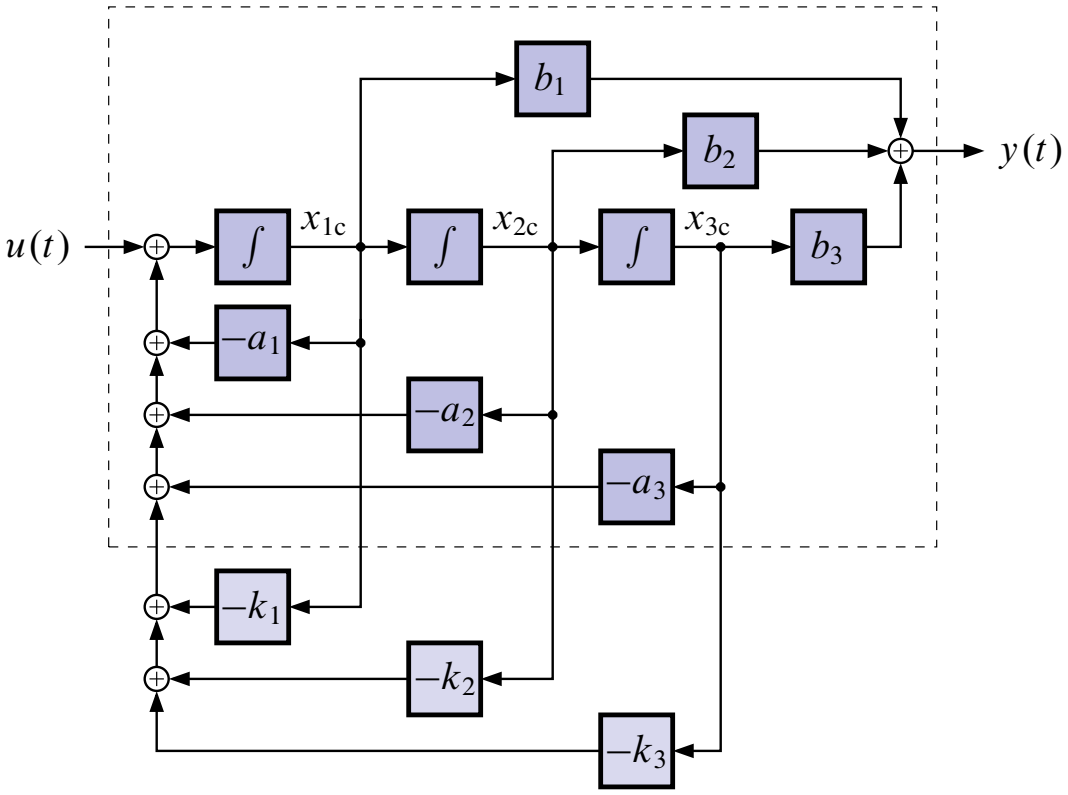
- Note, $K_c = \begin{bmatrix} k_1 & \cdots & k_n \end{bmatrix}$, so

$$B_c K_c = \begin{bmatrix} k_1 & k_2 & \cdots & k_n \\ 0 & & & 0 \\ \vdots & & & \vdots \\ 0 & & & 0 \end{bmatrix}.$$

- Useful because characteristic equation obvious.

$$A_{CL} = A_c - B_c K_c = \begin{bmatrix} -(a_1 + k_1) & -(a_2 + k_2) & \cdots & -(a_n + k_n) \\ 1 & & & 0 \\ & & \ddots & \vdots \\ 0 & & \cdots & 1 & 0 \end{bmatrix},$$

still in controller form!



- Thus, after state feedback with K_c the characteristic equation is

$$\det(sI - A_c + B_c K_c) = s^n + (a_1 + k_1)s^{n-1} + \cdots + (a_n + k_n).$$

- The desired characteristic equation is

$$\chi_d(s) = s^n + \alpha_1 s^{n-1} + \cdots + \alpha_n.$$

- Equating coefficients of powers-of-s, we set $k_1 = \alpha_1 - a_1, \ldots, k_n = \alpha_n - a_n$ and get the desired characteristic polynomial.

- Now that we have the solution in the controller canonical form, we transform back to the original realization

$$\begin{aligned}
\chi_d(s) &= \det(sI - A_c + B_c K_c) \\
&= \det(T)\det(sI - A_c + B_c K_c)\det(T^{-1}) \\
&= \det(sI - TA_cT^{-1} + TB_cK_cT^{-1}) \\
&= \det(sI - A + BK_cT^{-1}).
\end{aligned}$$

$$\chi(s) = \det(sI - A + BK) \qquad \text{so}$$

$$K = K_cT^{-1}.$$

So, if we use state feedback

$$\begin{aligned}
K &= K_cT^{-1} \\
&= \begin{bmatrix} (\alpha_1 - a_1) & \cdots & (\alpha_n - a_n) \end{bmatrix} T^{-1}
\end{aligned}$$

we will have the desired characteristic polynomial.

- One remaining question: What is T? We know $T = \mathcal{C}\mathcal{C}_c^{-1}$ and

$$\mathcal{C}_c = \begin{bmatrix} 1 & -a_1 & a_1^2 - a_2 & \cdots \\ 0 & 1 & -a_1 & 0 \\ \vdots & & 1 & \vdots \\ 0 & \cdots & & 1 \end{bmatrix}$$

$$\mathcal{C}_c^{-1} = \underbrace{\begin{bmatrix} 1 & a_1 & \cdots & a_{n-1} \\ 0 & 1 & & \vdots \\ \vdots & & \ddots & a_1 \\ 0 & & & 1 \end{bmatrix}}_{\text{upper } \Delta \text{ Toeplitz}}$$

$$T^{-1} = \begin{bmatrix} 1 & a_1 & \cdots & a_{n-1} \\ 0 & 1 & & \vdots \\ \vdots & & \ddots & a_1 \\ 0 & & & 1 \end{bmatrix}^{-1} \mathcal{C}^{-1}$$

- So,

$$K = \begin{bmatrix} (\alpha_1 - a_1) & \cdots & (\alpha_n - a_n) \end{bmatrix} \begin{bmatrix} 1 & a_1 & \cdots & a_{n-1} \\ 0 & 1 & & a_{n-2} \\ \vdots & & \ddots & \vdots \\ 0 & & & 1 \end{bmatrix}^{-1} \mathcal{C}^{-1}.$$

- This is called the *Bass–Gura formula* for K.

6.3: Ackermann's formula

- We need to know a_i to use the Bass–Gura formula. Ackermann's method may require less work.
- Consider (a system already in controller canonical form, for now),

$$\chi(s) = s^n + a_1 s^{n-1} + \cdots + a_n s^0$$

$$\chi(A_c) = A_c^n + a_1 A_c^{n-1} + \cdots + a_n I = 0$$

by Cayley–Hamilton.

- Also,

$$\chi_d(s) = s^n + \alpha_1 s^{n-1} + \cdots + \alpha_n s^0$$

$$\chi_d(A_c) = A_c^n + \alpha_1 A_c^{n-1} + \cdots + \alpha_n I \neq 0$$

$$\chi_d(A_c) - 0 = \chi_d(A_c) - \chi(A_c) = (\alpha_1 - a_1) A_c^{n-1} + \cdots + (\alpha_n - a_n) I.$$

- A sneaky fact for the controller form is,

$$\begin{bmatrix} 0 & \cdots & 1 \end{bmatrix} A_c^k = \begin{bmatrix} 0 & \cdots & \underbrace{1}_{n-k} & \cdots & 0 \end{bmatrix}$$

so that

$$\begin{bmatrix} 0 & \cdots & 1 \end{bmatrix} \chi_d(A_c) = \begin{bmatrix} (\alpha_1 - a_1) & (\alpha_2 - a_2) & \cdots & (\alpha_n - a_n) \end{bmatrix} = K_c$$

$$K_c = \begin{bmatrix} 0 & \cdots & 1 \end{bmatrix} \chi_d(A_c).$$

- To convert to the original form,

$$\begin{aligned} K &= K_c T^{-1} \\ &= \begin{bmatrix} 0 & \cdots & 1 \end{bmatrix} \chi_d(T^{-1} A T) T^{-1} \\ &= \begin{bmatrix} 0 & \cdots & 1 \end{bmatrix} T^{-1} \chi_d(A) \end{aligned}$$

$$= \begin{bmatrix} 0 & \cdots & 1 \end{bmatrix} \mathcal{C}_c \mathcal{C}^{-1} \chi_d(A)$$

$$= \begin{bmatrix} 0 & \cdots & 1 \end{bmatrix} \mathcal{C}^{-1} \chi_d(A).$$

- Revisit previous example. $\chi_d(s) = s^2 + 11s + 30$.

$$\mathcal{C} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$K = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \left\{ \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} + 11 \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} + 30 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right\}$$

$$= \begin{bmatrix} 0 & 1 \end{bmatrix} \left\{ \begin{bmatrix} 2 & 3 \\ 3 & 5 \end{bmatrix} + \begin{bmatrix} 41 & 11 \\ 11 & 52 \end{bmatrix} \right\}$$

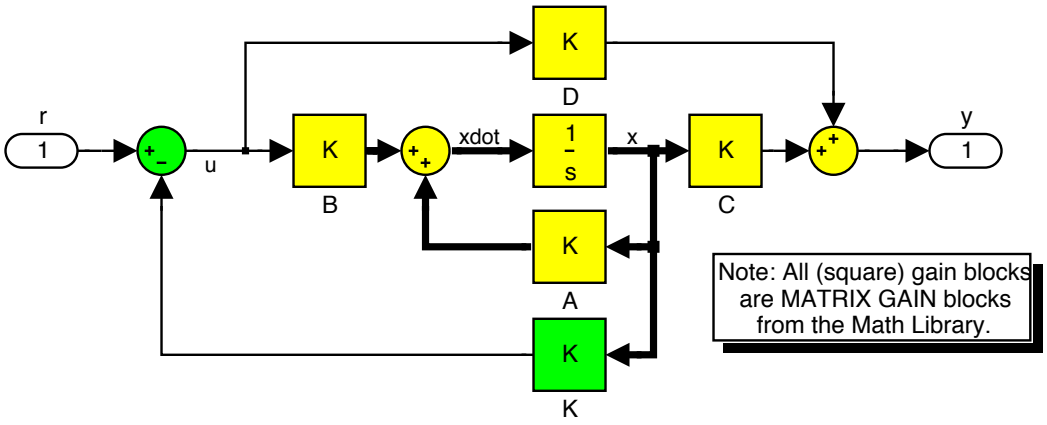
$$= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 43 & 14 \\ 14 & 57 \end{bmatrix}$$

$$= \begin{bmatrix} 14 & 57 \end{bmatrix}. \quad \checkmark \text{ same as before.}$$

- `K=acker(A,B,poles);` ➠ Very easy in Matlab, but numerical issues.
- `K=place(A,B,poles);` ➠ Use this instead, unless you have repeated roots.
- `polyvalm.m` ➠ To compute $\chi_d(A)$.

Simulating state feedback in Simulink

- The following block diagram may be used to simulate a state-feedback control system in Simulink.



Some comments

FACT: The eigenvalues associated with uncontrollable modes are fixed (don't change) under state feedback, but those associated with controllable modes can be arbitrarily assigned.

FACT: State feedback does not change zeros of a realization.

FACT: Drastic changes in characteristic polynomial requires large gains K (high control effort).

FACT: State feedback can result in unobservable modes (pole-zero cancellations).

6.4: Reference input

- So far, we have looked at how to pick K to get homogeneous dynamics that we want.

  $$\text{eig}(A_{CL}) \quad - \quad \text{fast/slow/real poles ...}$$

  How does this improve our ability to track a reference?

- Started with $u(t) = r(t) - Kx(t)$.
- Want $y(t) \approx r(t)$ for good tracking.
- Frequency domain, want $\dfrac{Y(s)}{R(s)} \approx 1$. Usually only get this performance at low frequencies.
- Problem is that $u(t) = r(t) - Kx(t)$ is simple, but it gives *steady-state errors.*

EXAMPLE:

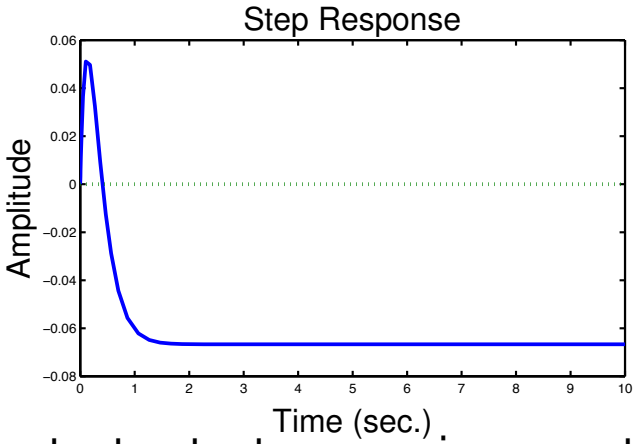
$$A = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad K = \begin{bmatrix} 14 & 57 \end{bmatrix}.$$

- Let $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$. Then $\dfrac{Y(s)}{R(s)} = C(sI - A + BK)^{-1}B$.

$$\frac{Y(s)}{R(s)} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s+13 & 56 \\ -1 & s-2 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \end{bmatrix} \frac{\begin{bmatrix} s-2 & -56 \\ 1 & s+13 \end{bmatrix}}{s^2 + 11s - 26 + 56} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{s-2}{s^2 + 11s + 30}.$$

- Final value theorem for step input, $y(t) \to \dfrac{-2}{30} \neq 1$!



OBSERVATION: A constant output y_{ss} requires constant state x_{ss} and constant input u_{ss}. We can change the tracking problem to a regulation problem around $u(t) = u_{ss}$ and $x(t) = x_{ss}$.

$$(u(t) - u_{ss}) = -K(x(t) - x_{ss}).$$

- u_{ss} and x_{ss} related to r_{ss}. Let

$$u_{ss} = \underbrace{N_u}_{1\times 1} r_{ss}$$
$$x_{ss} = \underbrace{N_x}_{n\times 1} r_{ss}.$$

- How to find N_u and N_x? Use equations of motion.

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t).$$

- At steady state,

$$\dot{x}_{ss}(t) = 0 = Ax_{ss} + Bu_{ss} = [AN_x + BN_u]r_{ss}$$
$$y_{ss}(t) = r_{ss} = Cx_{ss} + Du_{ss} = [CN_x + DN_u]r_{ss}.$$

- Two equations and two unknowns.

$$\left[\begin{array}{c:c} A & B \\ \hdashline C & D \end{array}\right]\left[\begin{array}{c} N_x \\ \hdashline N_u \end{array}\right] = \left[\begin{array}{c} 0 \\ \hdashline I \end{array}\right].$$

- In steady-state we had

$$(u(t) - u_{ss}) = -K(x(t) - x_{ss})$$

which is achieved by the control signal

$$\begin{aligned} u(t) &= N_u r(t) - K(x(t) - N_x r(t)) \\ &= -Kx(t) + (N_u + KN_x)r(t) \\ &= -Kx(t) + \bar{N}r(t). \end{aligned}$$

- $\bar{N}$ computed without knowing $r(t)$. It works for any $r(t)$.
- In our example we can find that

$$\left[\begin{array}{c} N_x \\ \hdashline N_u \end{array}\right] = \left[\begin{array}{c} 1 \\ -1/2 \\ \hdashline -1/2 \end{array}\right].$$

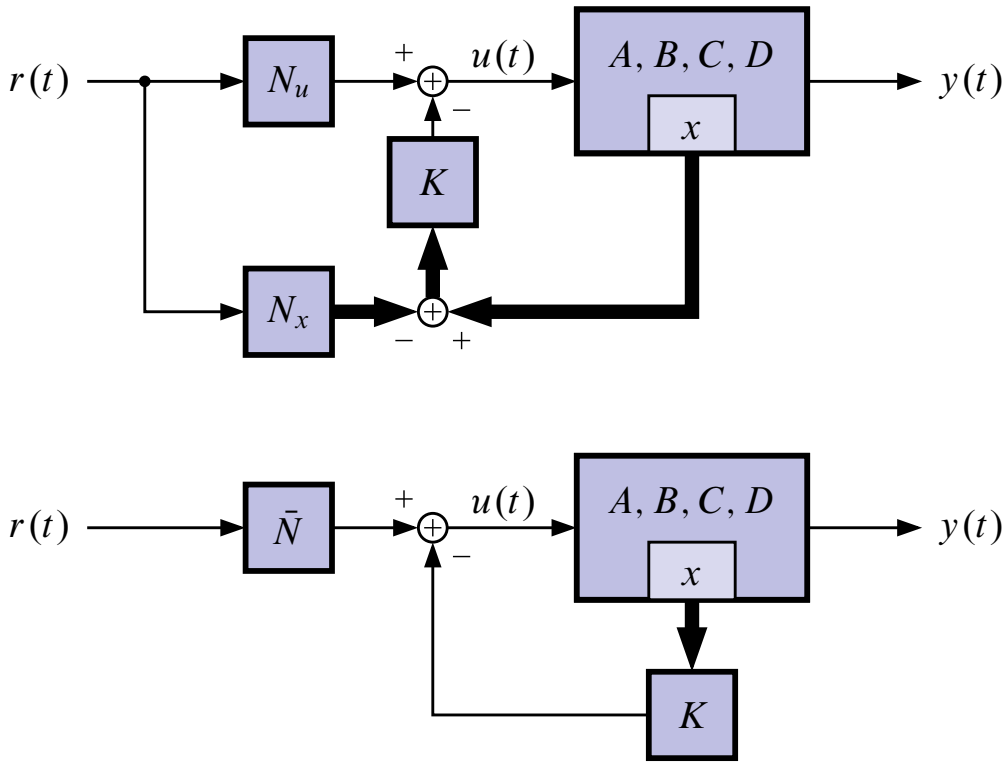
- $N_u + KN_x = -15$.
- New equations:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + B(N_u + KN_x)r(t) - BKx(t) \\ &= (A - BK)x(t) + B\bar{N}r(t) \\ y(t) &= Cx(t). \end{aligned}$$

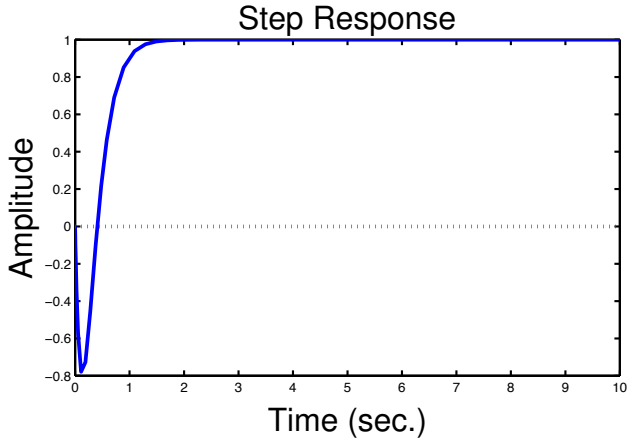
- Therefore,

$$\left(\frac{Y(s)}{R(s)}\right)_{\text{new}} = \left(\frac{Y(s)}{R(s)}\right)_{\text{old}} \times \bar{N} = \frac{-15s + 30}{s^2 + 11s + 30}$$

which has zero steady-state error to a unit-step.



- Simulate (either method)



6.5: Pole placement

- Classical question: Where do we place the closed-loop poles?

THOUGHT I: Dominant second-order behavior, just as before.

- Assume dominant behavior given by roots of

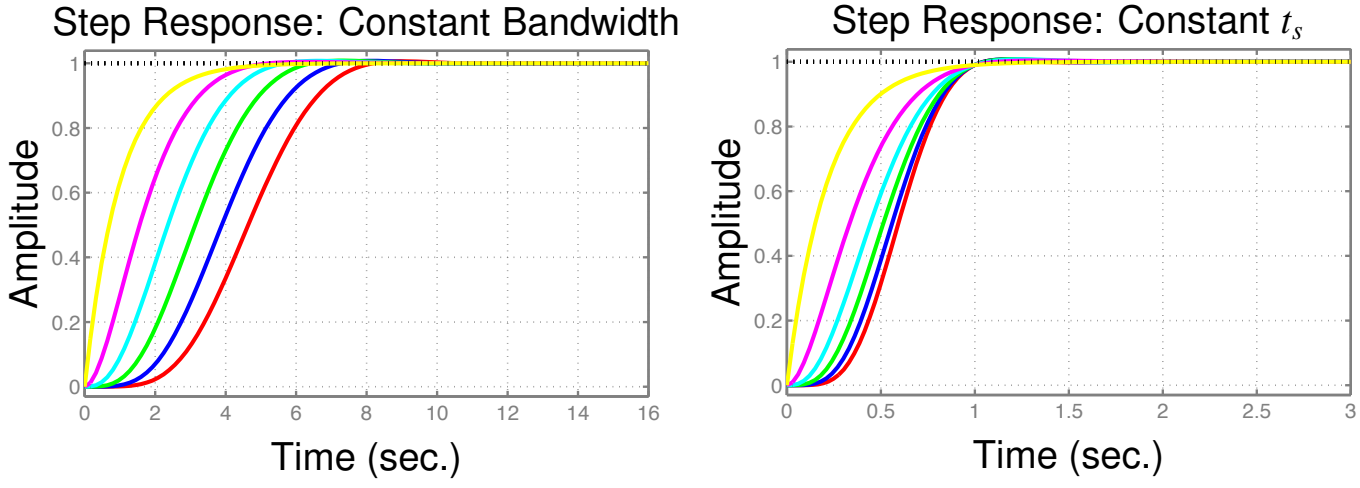
$$s^2 + 2\zeta\omega_n s + \omega_n^2 \quad \Rightarrow \quad s = -\left\{\omega_n \pm j\omega_n\sqrt{1-\zeta^2}\right\}$$

- Put other poles so that the time response is much faster than this dominant behavior.
- Place them so that they are "sufficiently damped."
  - Real part $< -4\zeta\omega_n$.
  - Keep frequency same as open loop.
- Be very careful about moving poles too far. Takes a lot of control effort.

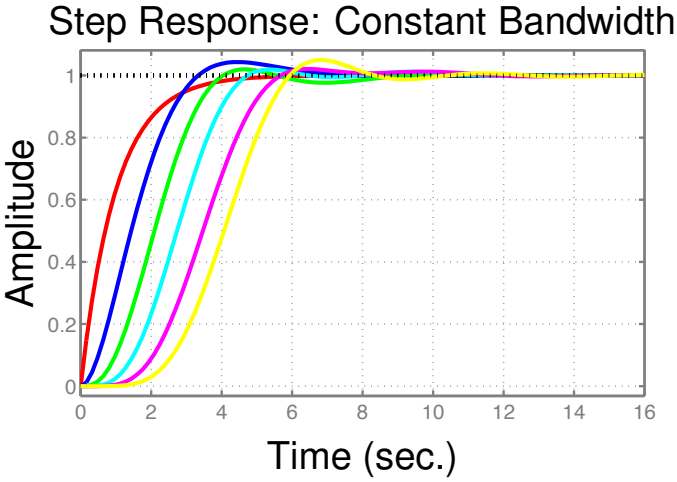
THOUGHT II: Can also choose closed-loop poles to mimic a system that has performance that you like. Set closed-loop poles equal to this prototype system.

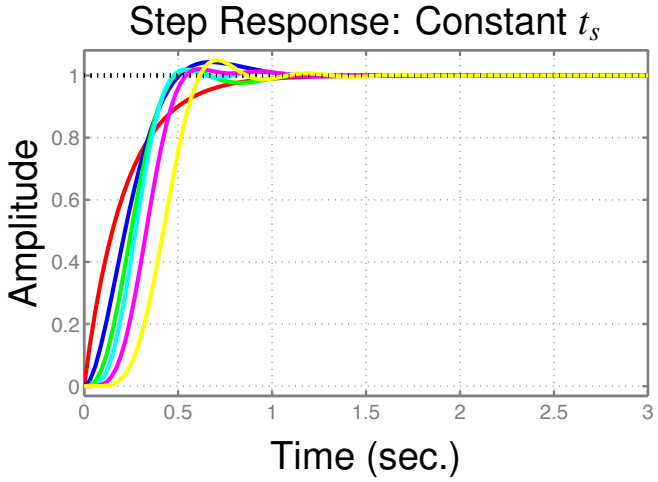
- Scaled to give settling time of 1 sec. or bandwidth of $\omega = 1$ rad/sec.

Bessel prototype systems



ITAE prototype systems





PROCEDURE: For nth-order system—desired bandwidth.

1. Determine desired bandwidth ω_o.
2. Find the nth-order poles from the table of constant bandwidth, and *multiply* pole locations by ω_o.
3. Use Acker/place to locate poles. Simulate and check control effort.

PROCEDURE: For nth-order system—desired settling time.

1. Determine desired settling time t_s.
2. Find the nth-order poles from the table of constant settling time, and *divide* pole locations by t_s.
3. Use Acker/place to locate poles. Simulate and check control effort.

- Bessel model has no overshoot, but is slow compared with ITAE.
- NOT a good idea for flexible systems. Why?

ITAE pole locations for $\omega_o = 1$ rad/sec.

1:	-1.000		
2:	$-0.707 \pm 0.707j$		
3:	$-0.521 \pm 1.068j$	-0.708	
4:	$-0.424 \pm 1.263j$	$-0.626 \pm 0.414j$	
5:	$-0.376 \pm 1.292j$	$-0.576 \pm 0.534j$	-0.896
6:	$-0.310 \pm 0.962j$	$-0.581 \pm 0.783j$	$-0.735 \pm 0.287j$

ITAE pole locations for $t_s = 1$ sec.

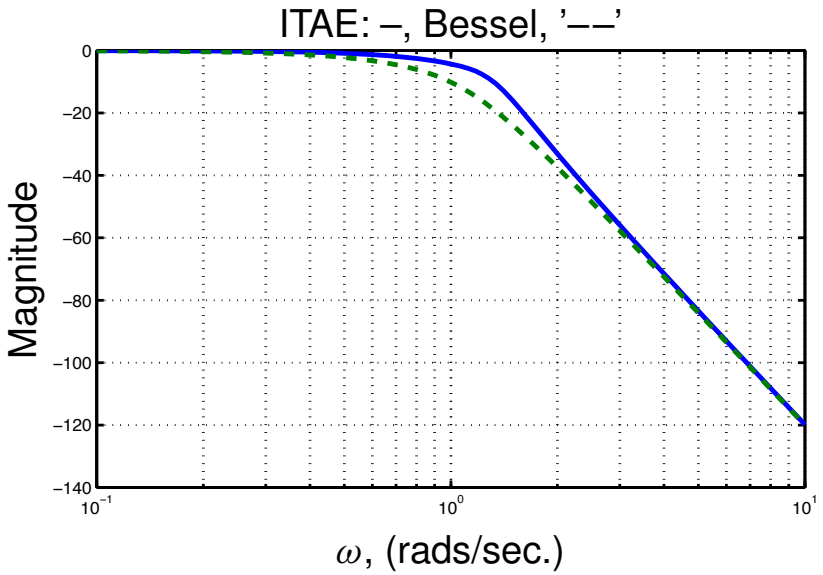
1:	-4.620		
2:	$-4.660 \pm 4.660j$		
3:	$-4.350 \pm 8.918j$	-5.913	
4:	$-4.236 \pm 12.617j$	$-6.254 \pm 4.139j$	
5:	$-3.948 \pm 13.553j$	$-6.040 \pm 5.601j$	-9.394
6:	$-2.990 \pm 12.192j$	$-5.602 \pm 7.554j$	$-7.089 \pm 2.772j$

Bessel pole locations for $\omega_o = 1$ rad/sec.

1:	-1.000		
2:	$-0.866 \pm 0.500j$		
3:	$-0.746 \pm 0.711j$	-0.942	
4:	$-0.657 \pm 0.830j$	$-0.905 \pm 0.271j$	
5:	$-0.591 \pm 0.907j$	$-0.852 \pm 0.443j$	-0.926
6:	$-0.539 \pm 0.962j$	$-0.800 \pm 0.562j$	$-0.909 \pm 0.186j$

Bessel pole locations for $t_s = 1$ sec.

1:	-4.620		
2:	$-4.053 \pm 2.340j$		
3:	$-3.967 \pm 3.785j$	-5.009	
4:	$-4.016 \pm 5.072j$	$-5.528 \pm 1.655j$	
5:	$-4.110 \pm 6.314j$	$-5.927 \pm 3.081j$	-6.448
6:	$-4.217 \pm 7.530j$	$-6.261 \pm 4.402j$	$-7.121 \pm 1.454j$



EXAMPLE:

$$G(s) = \frac{1}{s(s+1)(s+4)} \quad \Rightarrow \quad A = \begin{bmatrix} -5 & -4 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}.$$

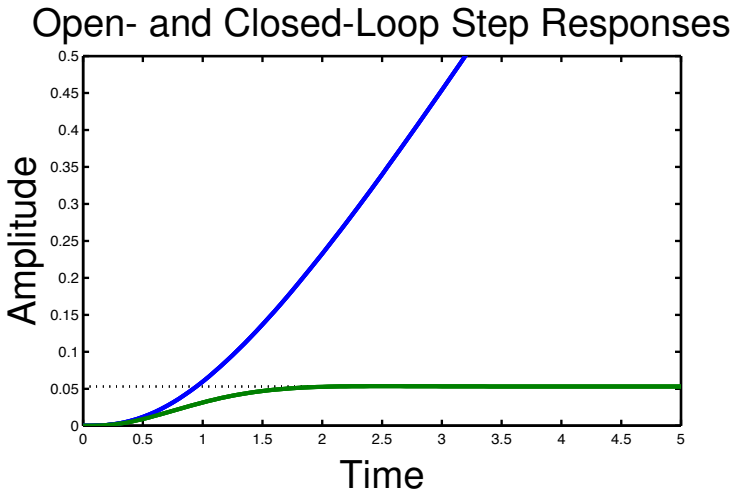
want $t_s = 2$, and 3rd-order Bessel.

$$s_1 = \frac{-5.009}{2} = -2.505$$

$$s_{2,3} = \frac{-3.967 \pm 3.784j}{2} = -1.984 \pm 1.892j$$

$$\chi_d(s) = s^3 + 6.473s^2 + 17.456s + 18.827.$$

$$K = \begin{bmatrix} 1.473 & 13.456 & 18.827 \end{bmatrix}.$$



- Poles are now in the right place, but poor steady-state error to step input. Use reference-tracking method from before

$$\left[\begin{array}{ccc:c} -5 & -4 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hdashline 0 & 0 & 1 & 0 \end{array}\right]\left[\begin{array}{c} N_x \\ \hdashline N_u \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \\ 0 \\ \hdashline 1 \end{array}\right].$$

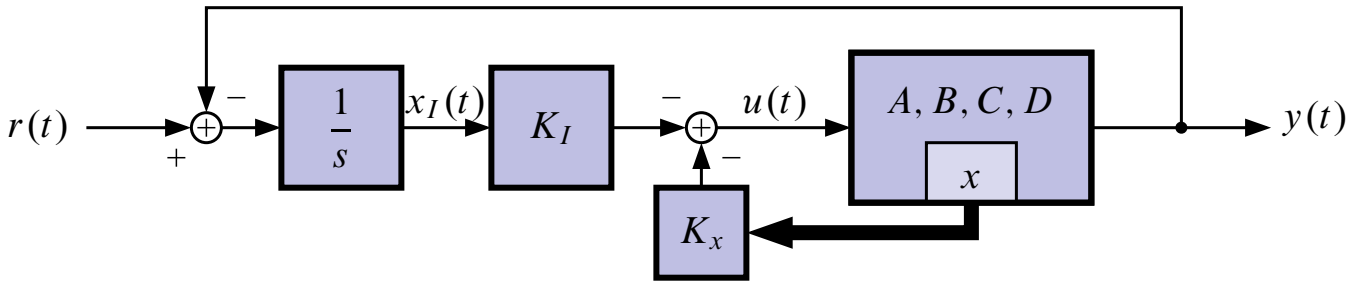
or

$$\left[\begin{array}{c} N_x \\ \hdashline N_u \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \\ 1 \\ \hdashline 0 \end{array}\right] \qquad \text{or} \qquad \bar{N} = 18.827.$$

- This "fixes" our step response, but we cannot use a similar strategy to improve ramp responses etc. Recall from ECE4510 that we need integrators in the open-loop system to increase system type.

6.6: Integral control for continuous-time systems

- In many practical designs, integral control is needed to counteract disturbances, plant variations, or other noises in the system.
- Up until now, we have not seen a design that has integral action. In fact state-space designs will NOT produce integral action unless we make special steps to include it!
- How do we introduce integral control? We augment our system with one or more integrators:



- In other words, include an integral state equation of

$$\begin{aligned}\dot{x}_I(t) &= r(t) - y(t) \\ &= r(t) - (Cx(t) + Du(t)).\end{aligned}$$

  and *THEN* design K_I and K_x such that the system had good closed-loop pole locations.

- Note that we can include the integral state into our normal state-space form by augmenting the system dynamics

$$\left[\begin{array}{c} \dot{x}_I(t) \\ \hdashline \dot{x}(t) \end{array}\right] = \left[\begin{array}{c:c} 0 & -C \\ \hdashline 0 & A \end{array}\right] \left[\begin{array}{c} x_I(t) \\ \hdashline x(t) \end{array}\right] + \left[\begin{array}{c} -D \\ \hdashline B \end{array}\right] u(t) + \left[\begin{array}{c} I \\ \hdashline 0 \end{array}\right] r(t)$$

$$y(t) = Cx(t) + Du(t).$$

- Note that the matrix that now fills the state-space placeholder usually called "A" has an open-loop eigenvalue at the origin. This corresponds to increasing the system type, and integrates out steady-state error.
- The control law is,

$$u(t) = -\left[\begin{array}{c:c} K_I & K_x \end{array}\right] \left[\begin{array}{c} x_I(t) \\ \hdashline x(t) \end{array}\right].$$

- This is state feedback on the augmented state vector. We can find the feedback gain vector K by using Ackerman's formula (or similar) by replacing "A" in Ackerman with the augmented "A" matrix above, and by replacing "B" in Ackerman by the matrix multiplying $u(t)$ above.
- Note that the augmented system has $n + n_I$ open-loop poles, so we will have to choose $n + n_I$ desired closed-loop poles, and split the resulting K into the parts K_I and K_x.
- When we substitute the control law for $u(t)$ into the open-loop augmented state-space dynamics, we get:

$$\left[\begin{array}{c} \dot{x}_I(t) \\ \hdashline \dot{x}(t) \end{array}\right] = \left(\overbrace{\underbrace{\left[\begin{array}{c:c} 0 & -C \\ \hdashline 0 & A \end{array}\right]}_{A \text{ in Acker/place}} - \underbrace{\left[\begin{array}{c} -D \\ \hdashline B \end{array}\right]}_{B \text{ in Acker/place}} \left[\begin{array}{c:c} K_I & K_x \end{array}\right]}^{A_{CL}} \right) \left[\begin{array}{c} x_I(t) \\ \hdashline x(t) \end{array}\right] + \left[\begin{array}{c} I \\ \hdashline 0 \end{array}\right] r(t)$$

- This is a state-space system with the augmented state vector, and input $r(t)$.
- Our previous example becomes:

$$\begin{bmatrix} \dot{x}_I(t) \\ \hdashline \dot{x}(t) \end{bmatrix} = \left[\begin{array}{c|ccc} 0 & 0 & 0 & -1 \\ \hdashline 0 & -5 & -4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \begin{bmatrix} x_I(t) \\ \hdashline x(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \hdashline 1 \\ 0 \\ 0 \end{bmatrix} u(t) + \begin{bmatrix} 1 \\ \hdashline 0 \\ 0 \\ 0 \end{bmatrix} r(t)$$

$$y(t) = \left[\begin{array}{c|ccc} 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} x_I(t) \\ \hdashline x(t) \end{bmatrix}.$$

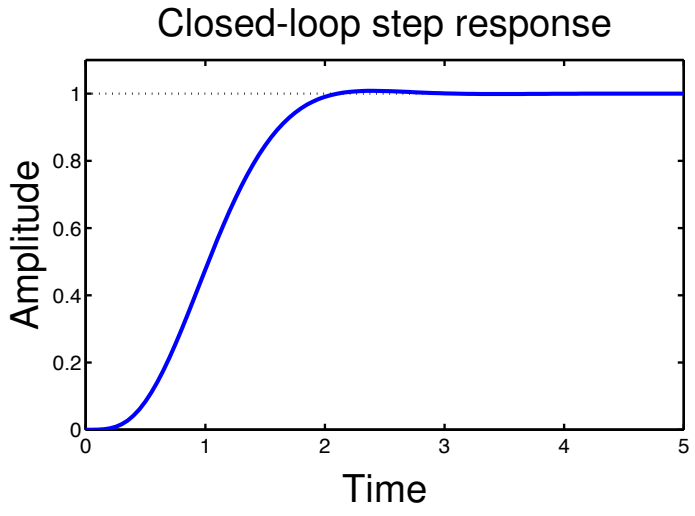
- We choose the fourth-order Bessel poles to give $t_s = 2.0$ seconds

$$s_{1,2} = (-4.016 \pm 5.072j)/2 \qquad \text{and} \qquad s_{3,4} = (-5.528 \pm 1.655j)/2.$$

- Using a pole-placement procedure, we find that

$$K = \begin{bmatrix} -87.102 & 4.544 & 36.988 & 91.273 \end{bmatrix}.$$

- Therefore, $K_I = -87.102$ and $K_x = \begin{bmatrix} 4.544 & 36.988 & 91.273 \end{bmatrix}$.
- When we simulate the closed-loop system (try it!) we get:



6.7: State feedback for discrete-time systems

- The result is identical.

  Characteristic frequencies of controllable modes are freely assignable by state feedback; characteristic frequencies of uncontrollable modes do not change with state feedback.

- There is a special characteristic polynomial for discrete-time systems

$$\chi(z) = z^n;$$

  that is, all eigenvalues are zero.

- What does this mean? By Cayley-Hamilton,

$$(A - BK)^n = 0.$$

- Hence, with no input, the state reaches 0 in at most n steps since

$$x[n] = (A - BK)^n x[0] = 0$$

  no matter what $x[0]$ is.

- This is called *dead-beat control* and $A - BK$ is called a *Nilpotent matrix.*

EXAMPLE: Consider

$$x[k+1] = \begin{bmatrix} 1 & 0 \\ 2 & 2 \end{bmatrix} x[k] + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u[k].$$

- This system is controllable, so we can find a $K = [k_1 \quad k_2]$ such that

$$\det \begin{bmatrix} z - 1 + k_1 & k_2 \\ -2 & z - 2 \end{bmatrix} = z^2$$

$$(z - 1 + k_1)(z - 2) + 2k_2 =$$

and therefore $k_1 = 3$ and $k_2 = 2$.

$$A - BK = \begin{bmatrix} -2 & -2 \\ 2 & 2 \end{bmatrix}$$

$$(A - BK)^2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \text{as claimed.}$$

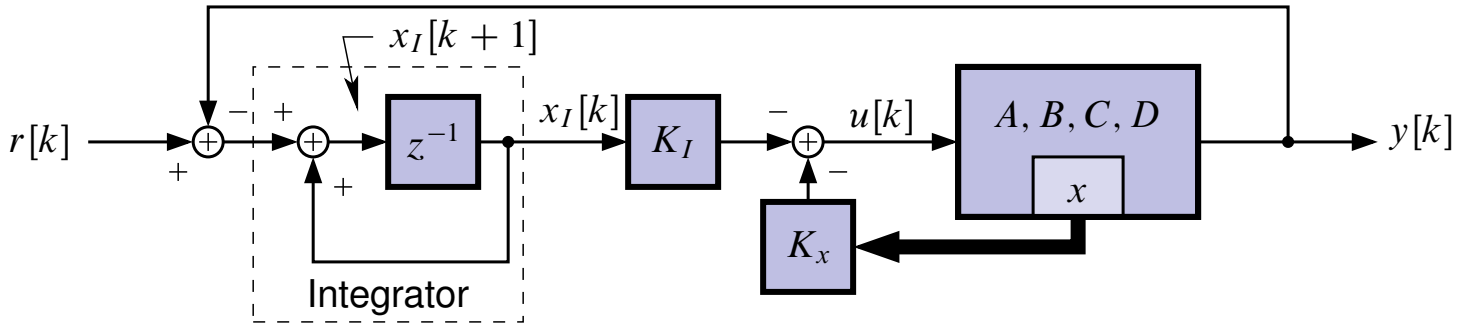
- The open-loop system is unstable, but the closed-loop system is not only stable but effects of initial conditions completely disappear after two time steps–they do not merely decay.
- This is a common design procedure, but beware of high control effort.

Reference input

- Tracking a reference input with a discrete-time system requires the same method as for continuous-time systems.

Integral control

- Again, we augment our system with a (discrete-time) integrator:



- In discrete time, we include an integral state equation of

$$\begin{aligned} x_I[k+1] &= x_I[k] + r[k] - y[k] \\ &= x_I[k] + r[k] - (Cx[k] + Du[k]). \end{aligned}$$

- We can include the integral state into our normal state-space form by augmenting the system dynamics

$$\begin{bmatrix} x_I[k+1] \\ \hdashline x[k+1] \end{bmatrix} = \overbrace{\left[\begin{array}{c:c} I & -C \\ \hdashline 0 & A \end{array}\right]}^{A \text{ in Acker/place}} \begin{bmatrix} x_I[k] \\ \hdashline x[k] \end{bmatrix} + \overbrace{\begin{bmatrix} -D \\ \hdashline B \end{bmatrix}}^{B \text{ in Acker/place}} u[k] + \begin{bmatrix} I \\ \hdashline 0 \end{bmatrix} r[k]$$

$$y[k] = Cx[k] + Du[k].$$

- Notice the new open-loop eigenvalue of "A" at $z = 1$.
- The control law is,

$$u[k] = -\left[\begin{array}{c:c} K_I & K_x \end{array}\right] \begin{bmatrix} x_I[k] \\ \hdashline x[k] \end{bmatrix}.$$

- So, again, we now have the task of choosing $n + n_I$ closed-loop poles.

Discrete-time prototype pole placement

- Where do we place the closed-loop poles?
- Can choose closed-loop poles to mimic a system that has performance that you like. Set closed-loop poles equal to this prototype system.
- Can be done using the ITAE and Bessel (continuous-time) tables.

PROCEDURE: For nth-order system—desired bandwidth.

1. Determine desired bandwidth ω_o.
2. Find the nth-order poles from the table of constant bandwidth, and *multiply* pole locations by ω_o.
3. *Convert s-plane locations to z-plane locations using $z = e^{sT}$.*

4. Use Acker/place to locate poles. Simulate and check control effort.

PROCEDURE: For nth-order system—desired settling time.

1. Determine desired settling time t_s.
2. Find the nth-order poles from the table of constant settling time, and *divide* pole locations by t_s.
3. *Convert s-plane locations to z-plane locations using $z = e^{sT}$.*
4. Use Acker/place to locate poles. Simulate and check control effort.

6.8: MIMO control design

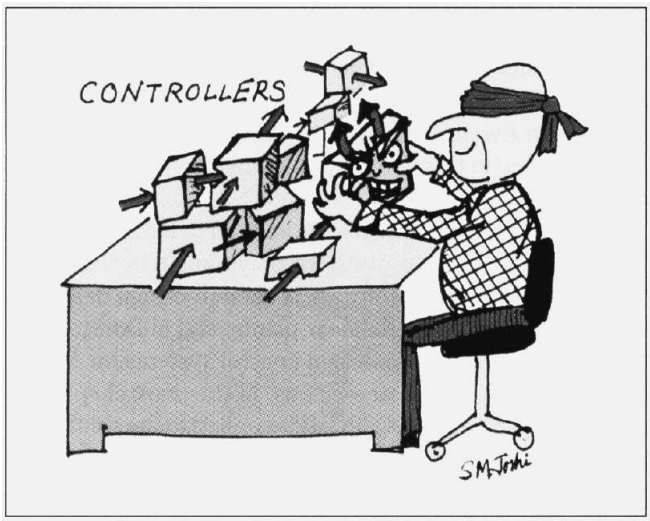
- So far, we have discussed control design for SISO systems only.
- Several different MIMO approaches exist, and all require finding K such that $u(t) = -Kx(t)$.
- K has as many rows as $u(t)$, as many columns as there are states.

FACT: If a MIMO system is controllable, it is possible to choose a K matrix to place the poles of the system anywhere in the s-plane (or z-plane) in complex-conjugate pairs.

FACT: If a MIMO system is controllable, the matrix K ***is not unique***! This brings up the question of optimal values of K...

- A number of design approaches exist. Some are very methodical, but difficult. Others employ randomness but are easier (if they work).

The symbolic computation method: Sometimes a long wait!



The probabilistic method: Guess, it's probably O.K.

- We will investigate two of the "random" methods.

Cyclic design

- Cyclic design changes multi-input problem to a single-input problem.

- A matrix is *cyclic* if it has one and only one Jordan block associated with each distinct eigenvalue. Note, this does not imply that all eigenvalues are distinct. (Although, if all eigenvalues are distinct, the matrix is cyclic).
- If the n-dimensional p-input pair $\{A, B\}$ is controllable and if A is cyclic, then for almost any $p \times 1$ vector v, the single-input pair $\{A, Bv\}$ is controllable.
  - Controllability is invariant under any equivalence transformation; thus, we may assume A to be in Jordan form. For example,

$$A = \begin{bmatrix} 2 & 1 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 2 \\ 4 & 3 \\ 1 & 0 \end{bmatrix}; \quad Bv = B \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \times \\ \times \\ \alpha \\ \times \\ \beta \end{bmatrix}$$

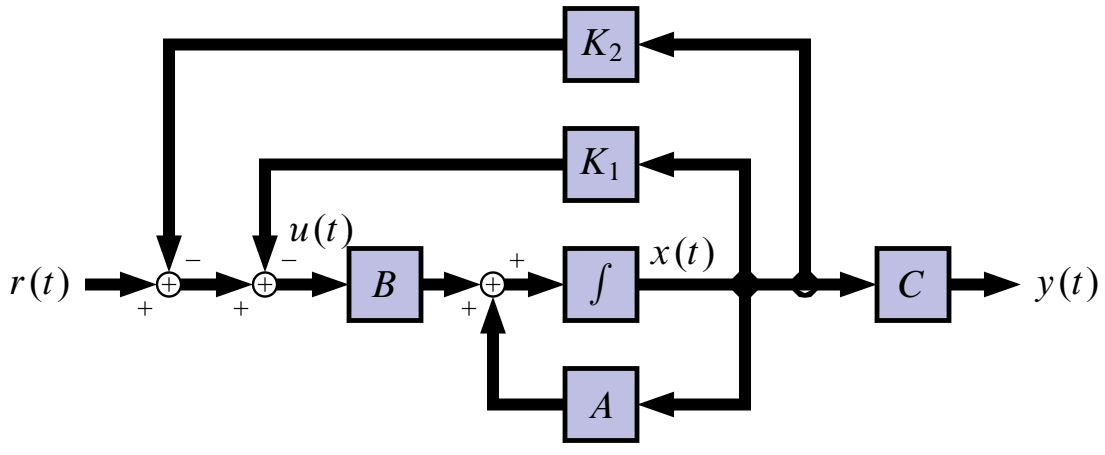
  - It can be shown that the condition for controllability is that $\alpha \neq 0$ and $\beta \neq 0$. Because $\alpha = v_1 + 2v_2$ and $\beta = v_1$ the system is uncontrollable only if $v_1 = 0$ or $v_1/v_2 = -2$.
  - Any other v vector in two-dimensional space results in a controllable pair $\{A, Bv\}$. If we choose v randomly, then "with probability 1," we will choose a good one.
- Design method:
  1. Randomly choose a vector v. Test for controllability of $\{A, Bv\}$. Repeat until controllable.
  2. The multi-input system $\{A, B, C, D\}$ has been reduced to a single-input system by stating that $u(t) = vu'(t)$. The new system

is $\{A, Bv, C, D\}$ with input $u'(t)$ and output $y(t)$. Use single-input design methods such as Bass–Gura or Ackermann to find k to place the poles of the single-input system. Then, the overall state feedback is: $u(t) = vu'(t)$ and $u'(t) = -kx(t)$ so $u(t) = -vkx(t)$. Therefore, $K = vk$.

- What if A is not cyclic? It can be shown that if $\{A, B\}$ is controllable then for almost any $p \times n$ real constant matrix K_1 the matrix $(A - BK_1)$ has only distinct eigenvalues, and is therefore cyclic. Randomly choose K_1 matrices until $(A - BK_1)$ is cyclic. Then, design for this system.
- Use small random numbers for low control effort.

DESIGN METHOD SUMMARY:

1. First, randomly choose $p \times n$ constant matrix K_1 such that $\bar{A} = A - BK_1$ is cyclic.
2. Randomly choose $p \times 1$ vector v such that $\{\bar{A}, Bv\}$ is controllable.
3. Design state feedback vector k using Bass–Gura/ Ackermann/ etc on system $\{\bar{A}, Bv\}$ to put the poles in the desired place. Then $K_2 = vk$. Assemble together



4. Design may be summed up as $u(t) = r(t) - (K_1 + K_2)x(t)$.

Lyapunov-equation design

DESIGN METHOD:

1. Select an $n \times n$ matrix F with a set of desired eigenvalues that contain no eigenvalues of A.
   - Make F real! If you have desired complex modes, use the real modal form.
   - Don't repeat eigenvalues in a diagonal matrix or $\{F, \bar{K}\}$ will not be observable. Use a Jordan form for repeated desired modes.
2. Randomly select $p \times n$ matrix $\bar{K}$ such that $\{F, \bar{K}\}$ is "observable."
3. Solve for the unique T in the Lyapunov equation $AT - TF = B\bar{K}$.
4. If T is singular, select a different $\bar{K}$ and repeat. If T is nonsingular, we compute $K = \bar{K}T^{-1}$ and $(A - BK)$ has the desired eigenvalues.

- If T is nonsingular, the Lyapunov equation and $KT = \bar{K}$ imply

$$(A - BK)T = TF \quad \text{or} \quad A - BK = TFT^{-1}.$$

- So, $(A - BK)$ and F are similar and have same set of eigenvalues.

Where to from here?

- In the design of state-feedback control, we assumed that all states of our plant were measured.
- This is often *impossible* to do or *too expensive*.
- So, we now investigate methods of reconstructing the plant state vector given only limited measurements.