Topic #11

16.31 Feedback Control Systems

State-Space Systems

- **Full-state Feedback Control**
- How do we change the poles of the state-space system?
- Or, even if we can change the pole locations.
- Where do we change the pole locations to?
- How well does this approach work?

- Reading: FPE 7.3

Full-state Feedback Controller

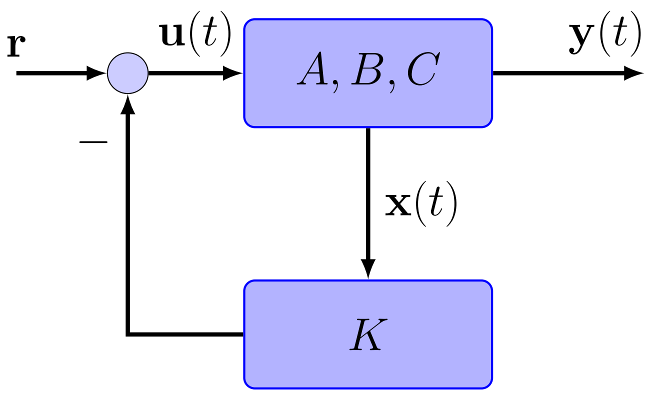
- Assume that the single-input system dynamics are given by

$$\begin{aligned}\dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + B\mathbf{u}(t)\\ \mathbf{y}(t) &= C\mathbf{x}(t)\end{aligned}$$

so that $D = 0$.

  - The multi-actuator case is quite a bit more complicated as we would have many extra degrees of freedom.

- Recall that the system poles are given by the eigenvalues of A.
  - Want to use the input $\mathbf{u}(t)$ to modify the eigenvalues of A to change the system dynamics.



- Assume a full-state feedback of the form:

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

where $\mathbf{r}$ is some **reference input** and the **gain** K is $\mathbb{R}^{1\times n}$

  - If $\mathbf{r} = 0$, we call this controller a **regulator**

- Find the closed-loop dynamics:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + B(\mathbf{r} - K\mathbf{x}(t))\\ &= (A - BK)\mathbf{x}(t) + B\mathbf{r}\\ &= A_{cl}\mathbf{x}(t) + B\mathbf{r}\\ \mathbf{y}(t) &= C\mathbf{x}(t)\end{aligned}$$

- **Objective:** Pick K so that A_{cl} has the desired properties, *e.g.*,
  - A unstable, want A_{cl} stable
  - Put 2 poles at $-2 \pm 2\mathbf{i}$
- Note that there are n parameters in K and n eigenvalues in A, so it looks promising, but what can we achieve?

- **Example #1:** Consider:

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

  - Then $\det(sI - A) = (s-1)(s-2) - 1 = s^2 - 3s + 1 = 0$ so the system is unstable.

  - Define $u = -\begin{bmatrix} k_1 & k_2 \end{bmatrix} \mathbf{x}(t) = -K\mathbf{x}(t)$, then

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

    which gives

$$\det(sI - A_{cl}) = s^2 + (k_1 - 3)s + (1 - 2k_1 + k_2) = 0$$

  - Thus, by choosing k_1 and k_2, we can put $\lambda_i(A_{cl})$ anywhere in the complex plane (assuming complex conjugate pairs of poles).

- To put the poles at $s = -5, \ -6$, compare the *desired characteristic equation*

$$(s+5)(s+6) = s^2 + 11s + 30 = 0$$

with the closed-loop one

$$s^2 + (k_1 - 3)s + (1 - 2k_1 + k_2) = 0$$

to conclude that

$$\left.\begin{array}{r} k_1 - 3 = 11 \\ 1 - 2k_1 + k_2 = 30 \end{array}\right\} \quad \begin{array}{l} k_1 = 14 \\ k_2 = 57 \end{array}$$

so that $K = \left[\ 14 \ \ 57\ \right]$, which is called **Pole Placement**.

- Of course, it is not always this easy, as lack of **controllability** might be an issue.

- **Example #2:** Consider this system:

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

with the same control approach

$$A_{cl} = A - BK = \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} \left[\ k_1 \ \ k_2\ \right] = \begin{bmatrix} 1 - k_1 & 1 - k_2 \\ 0 & 2 \end{bmatrix}$$

so that

$$\det(sI - A_{cl}) = (s - 1 + k_1)(s - 2) = 0$$

So the feedback control can modify the pole at $s = 1$, but it cannot move the pole at $s = 2$.

- **System cannot be stabilized with full-state feedback.**
- Problem caused by a lack of controllability of the e^{2t} mode.

- Consider the basic controllability test:

$$\mathcal{M}_c = \left[\, B \,\middle|\, AB \,\right] = \left[\begin{bmatrix} 1 \\ 0 \end{bmatrix} \middle| \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right]$$

So that $\texttt{rank}\ \mathcal{M}_c = 1 < 2$.

- **Modal analysis** of controllability to develop a little more insight

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix}, \quad \text{decompose as} \quad AV = V\Lambda \quad \Rightarrow \Lambda = V^{-1}AV$$

where

$$\Lambda = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \qquad V = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \qquad V^{-1} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$$

Convert

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + Bu \quad \xrightarrow{z=V^{-1}\mathbf{x}(t)} \quad \dot{z} = \Lambda z + V^{-1}Bu$$

where $z = \left[\ z_1 \quad z_2\ \right]^T$. But:

$$V^{-1}B = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix}$$

so that the dynamics in modal form are:

$$\dot{z} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} z + \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} u$$

- With this zero in the modal B-matrix, can easily see that the mode associated with the z_2 state is **uncontrollable.**
  - **Must assume that the pair $(A,\ B)$ are controllable.**

Ackermann's Formula

- The previous outlined a design procedure and showed how to do it by hand for second-order systems.
  - Extends to higher order (controllable) systems, but tedious.

- **Ackermann's Formula** gives us a method of doing this entire design process is one easy step.

$$K = \begin{bmatrix} 0 & \ldots & 0 & 1 \end{bmatrix} \mathcal{M}_c^{-1} \Phi_d(A)$$

  - $\mathcal{M}_c = \begin{bmatrix} B & AB & \ldots & A^{n-1}B \end{bmatrix}$ as before
  - $\Phi_d(s)$ is the characteristic equation for the closed-loop poles, which we then evaluate for $s = A$.
  - Note: is explicit that the **system must be controllable** because we are inverting the controllability matrix.

- Revisit **Example # 1:** $\Phi_d(s) = s^2 + 11s + 30$

$$\mathcal{M}_c = \left[B \,|\, AB \right] = \left[\begin{bmatrix} 1 \\ 0 \end{bmatrix} \middle| \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right] = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

  So

$$\begin{aligned} K &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}^{-1} \left(\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}^2 + 11 \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} + 30I \right) \\ &= \begin{bmatrix} 0 & 1 \end{bmatrix} \left(\begin{bmatrix} 43 & 14 \\ 14 & 57 \end{bmatrix} \right) = \begin{bmatrix} 14 & 57 \end{bmatrix} \end{aligned}$$

- Automated in Matlab: `place.m` & `acker.m` (see `polyvalm.m` too)

Origins of Ackermann's Formula

- For simplicity, consider third-order system (case #2 on 6–??), but this extends to any order.

$$A = \begin{bmatrix} -a_1 & -a_2 & -a_3 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad C = \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$$

  - See key benefit of using **control canonical** state-space model
  - This form is useful because the characteristic equation for the system is obvious $\Rightarrow \det(sI - A) = s^3 + a_1 s^2 + a_2 s + a_3 = 0$

- Can show that

$$\begin{aligned} A_{cl} = A - BK &= \begin{bmatrix} -a_1 & -a_2 & -a_3 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix} \\ &= \begin{bmatrix} -a_1 - k_1 & -a_2 - k_2 & -a_3 - k_3 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \end{aligned}$$

  so that the characteristic equation for the system is still obvious:

$$\begin{aligned} \Phi_{cl}(s) &= \det(sI - A_{cl}) \\ &= s^3 + (a_1 + k_1)s^2 + (a_2 + k_2)s + (a_3 + k_3) = 0 \end{aligned}$$

- Compare with the characteristic equation developed from the desired closed-loop pole locations:

$$\Phi_d(s) = s^3 + (\alpha_1)s^2 + (\alpha_2)s + (\alpha_3) = 0$$

to get that

$$\left.\begin{array}{c} a_1 + k_1 = \alpha_1 \\ \vdots \\ a_n + k_n = \alpha_n \end{array}\right\} \begin{array}{c} k_1 = \alpha_1 - a_1 \\ \vdots \\ k_n = \alpha_n - a_n \end{array}$$

- To get the specifics of the Ackermann formula, we then:
  - Take an arbitrary A, B and transform it to the control canonical form ($\mathbf{x}(t) \rightsquigarrow \mathbf{z}(t) = T^{-1}\mathbf{x}(t)$)
    - ♦ Not obvious, but $\mathcal{M}_c$ can be used to form this T
  - Solve for the gains $\hat{K}$ using the formulas at top of page for the state $\mathbf{z}(t)$

$$u(t) = \hat{K}\mathbf{z}(t)$$

  - Then switch back to gains needed for the state $\mathbf{x}(t)$, so that

$$K = \hat{K}T^{-1} \Rightarrow u = \hat{K}\mathbf{z}(t) = K\mathbf{x}(t)$$

- Pole placement is a very powerful tool and we will be using it for most of this course.

Reference Inputs

- So far we have looked at how to pick K to get the dynamics to have some nice properties (*i.e.* stabilize A)

- The question remains as to how well this controller allows us to track a reference command?
  - Performance issue rather than just stability.

- Started with

$$\begin{aligned} \dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + Bu \qquad y = C\mathbf{x}(t) \\ u &= r - K\mathbf{x}(t) \end{aligned}$$

- For **good tracking performance** we want

$$y(t) \approx r(t) \text{ as } t \to \infty$$

- Consider this performance issue in the frequency domain. Use the final value theorem:

$$\lim_{t\to\infty} y(t) = \lim_{s\to 0} sY(s)$$

  Thus, for good performance, we want

$$sY(s) \approx sR(s) \text{ as } s \to 0 \quad \Rightarrow \quad \left.\frac{Y(s)}{R(s)}\right|_{s=0} = 1$$

- So, for good performance, the transfer function from $R(s)$ to $Y(s)$ should be approximately 1 at DC.

- **Example #1 continued:** For the system

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

  - Already designed $K = \begin{bmatrix} 14 & 57 \end{bmatrix}$ so the closed-loop system is

$$\begin{aligned}\dot{\mathbf{x}}(t) &= (A - BK)\mathbf{x}(t) + Br \\ y &= C\mathbf{x}(t)\end{aligned}$$

  which gives the transfer function

$$\begin{aligned}\frac{Y(s)}{R(s)} &= C\left(sI - (A - BK)\right)^{-1} B \\ &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s+13 & 56 \\ -1 & s-2 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ &= \frac{s-2}{s^2 + 11s + 30}\end{aligned}$$

- Assume that $r(t)$ is a step, then by the FVT

$$\left.\frac{Y(s)}{R(s)}\right|_{s=0} = -\frac{2}{30} \neq 1 \text{ !!}$$

  - So our step response is quite poor!

- One solution is to scale the reference input $r(t)$ so that

$$u = \overline{N}r - K\mathbf{x}(t)$$

  - $\overline{N}$ extra gain used to scale the closed-loop transfer function

- Now we have

$$\begin{aligned} \dot{\mathbf{x}}(t) &= (A - BK)\mathbf{x}(t) + B\overline{N}r \\ y &= C\mathbf{x}(t) \end{aligned}$$

so that

$$\frac{Y(s)}{R(s)} = C\left(sI - (A - BK)\right)^{-1} B\overline{N} = G_{cl}(s)\overline{N}$$

If we had made $\overline{N} = -15$, then

$$\frac{Y(s)}{R(s)} = \frac{-15(s-2)}{s^2 + 11s + 30}$$

so with a step input, $y(t) \to 1$ as $t \to \infty$.

- Clearly can compute

$$\overline{N} = G_{cl}(0)^{-1} = -\left(C(A - BK)^{-1}B\right)^{-1}$$

- Note that this development assumed that r was constant, but it could also be used if r is a slowly time-varying command.

- So the steady state step error is now zero, but is this OK?
  - See plots – big improvement in the response, but transient a bit weird.

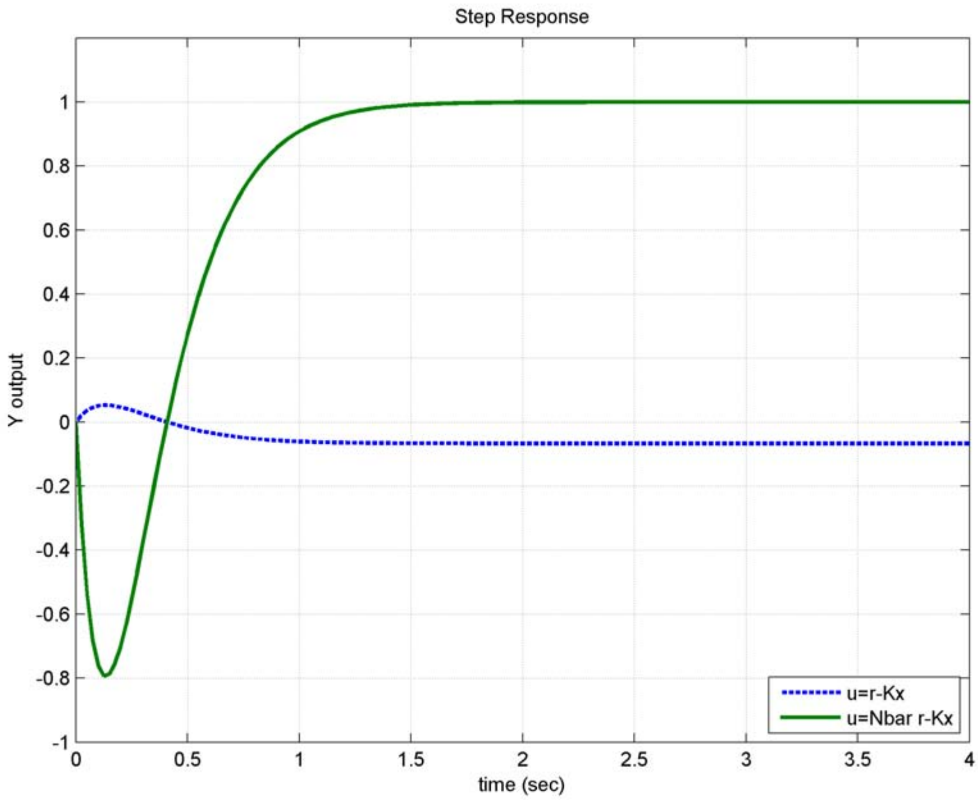


Fig. 1: Response to step input with and without the $\overline{N}$ correction.

Code: Step Response (step1.m)

```
% full state feedback for topic 13
% reference input issues
%
a=[1 1;1 2];b=[1 0]';c=[1 0];d=0;
k=[14 57];
Nbar=-15;
sys1=ss(a-b*k,b,c,d);
sys2=ss(a-b*k,b*Nbar,c,d);
t=[0:.025:4];
[y,t,x]=step(sys1,t);
[y2,t2,x2]=step(sys2,t);

plot(t,y,'--',t2,y2,'LineWidth',2);axis([0 4 -1 1.2]);grid;
legend('u=r-Kx','u=Nbar r-Kx','Location','SouthEast')
xlabel('time (sec)');ylabel('Y output');title('Step Response')
print -dpng -r300 step1.png
```

Pole Placement Examples

- Simple example:

$$G(s) = \frac{8 \cdot 14 \cdot 20}{(s+8)(s+14)(s+20)}$$

  - Target pole locations $-12 \pm 12\mathbf{i}$, -20

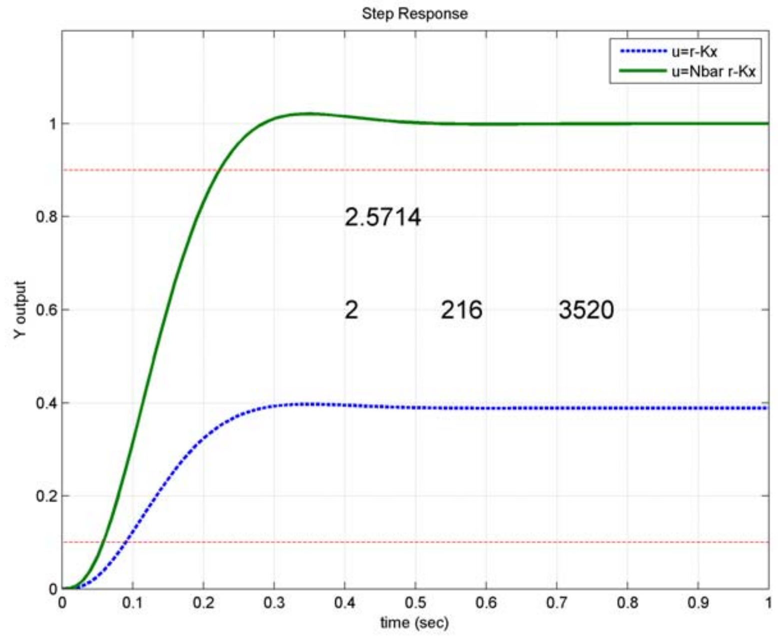


Fig. 2: Response to step input with and without the $\overline{N}$ correction. Gives the desired steady-state behavior, with little difficulty!

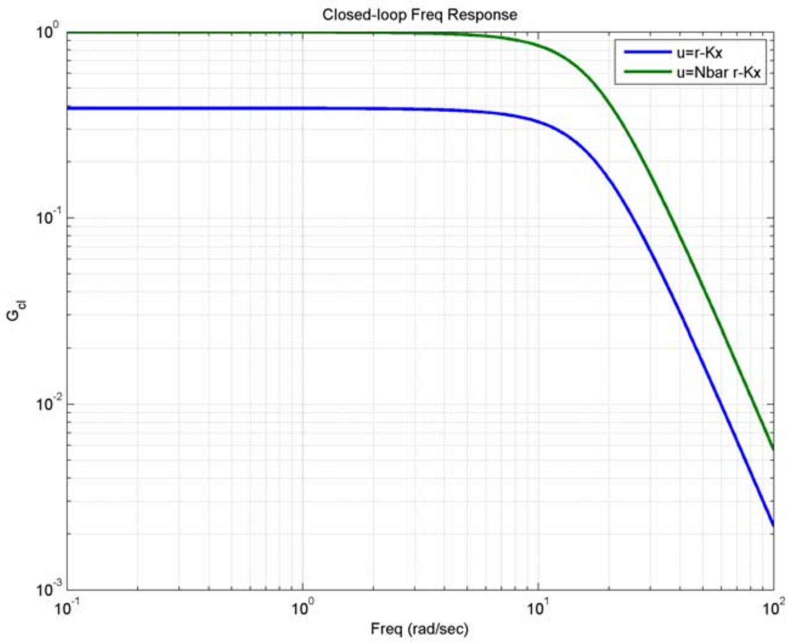


Fig. 3: Closed-loop frequency response. Clearly shows unity DC gain

- Example system with 1 unstable pole

$$G(s) = \frac{0.94}{s^2 - 0.0297}$$

  - Target pole locations $-0.25 \pm 0.25\mathbf{i}$

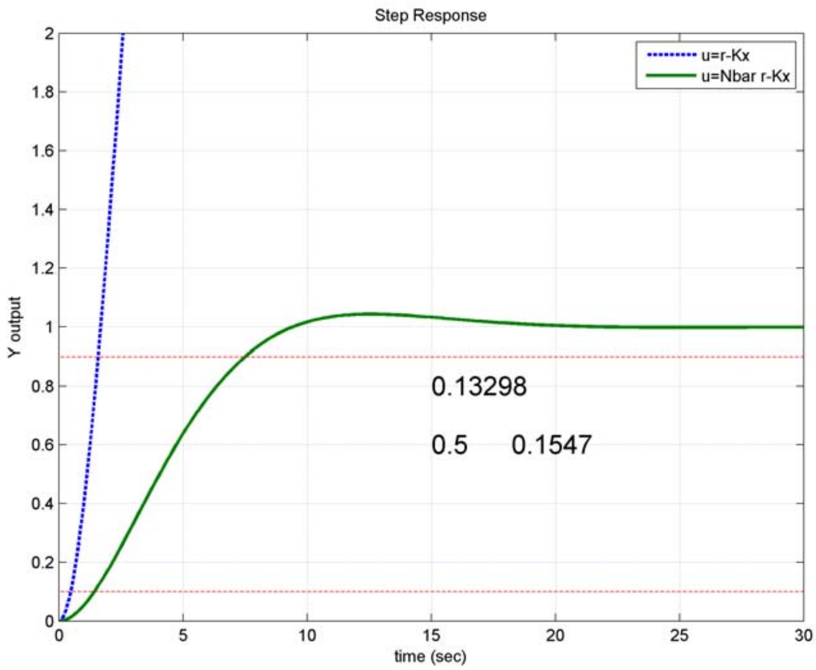


Fig. 4: Response to step input with and without the $\overline{N}$ correction. Gives the desired steady-state behavior, with little difficulty!

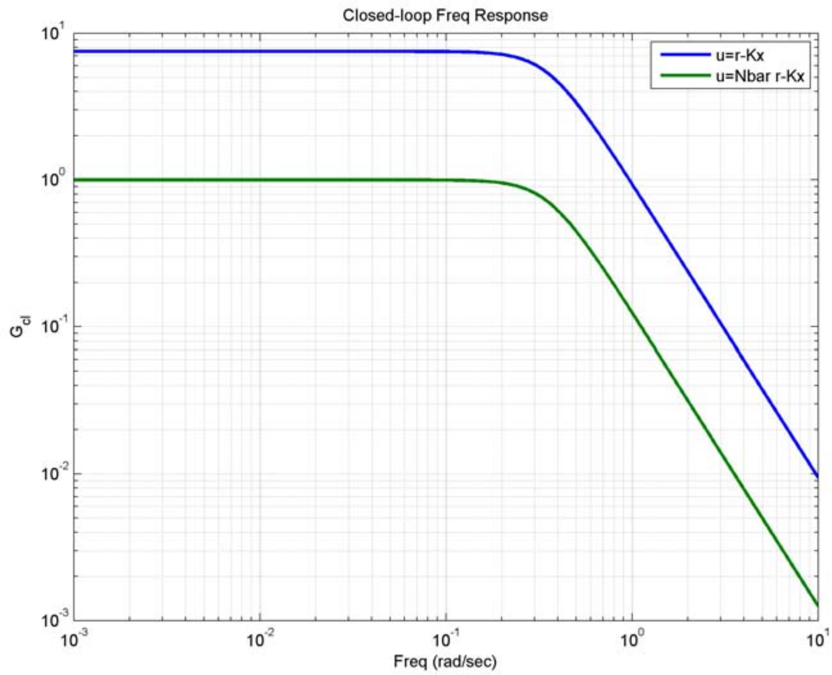


Fig. 5: Closed-loop frequency response. Clearly shows unity DC gain

- OK, so let's try something challenging...

$$G(s) = \frac{8 \cdot 14 \cdot 20}{(s-8)(s-14)(s-20)}$$

  - Target pole locations $-12 \pm 12\mathbf{i}$, -20

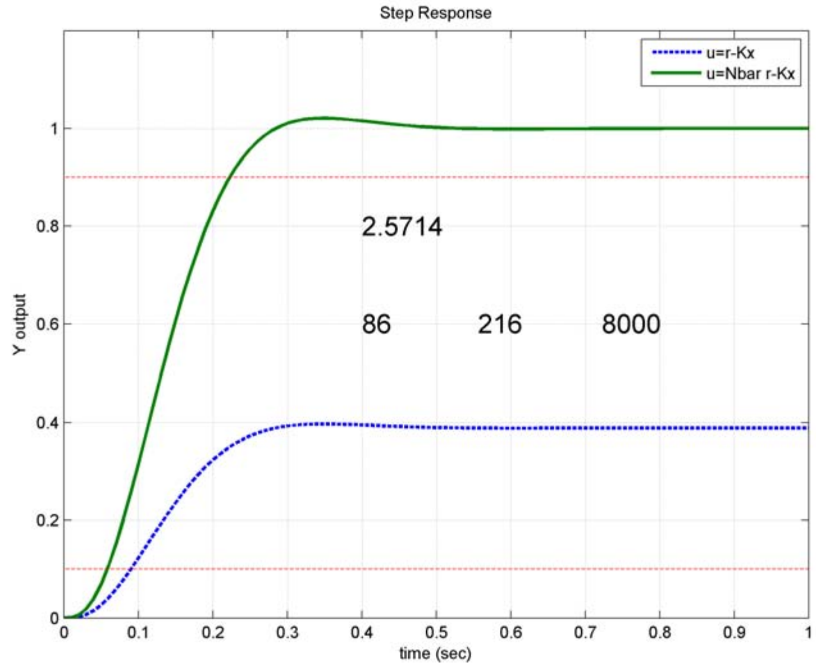


Fig. 6: Response to step input with and without the $\overline{N}$ correction. Gives the desired steady-state behavior, with little difficulty!

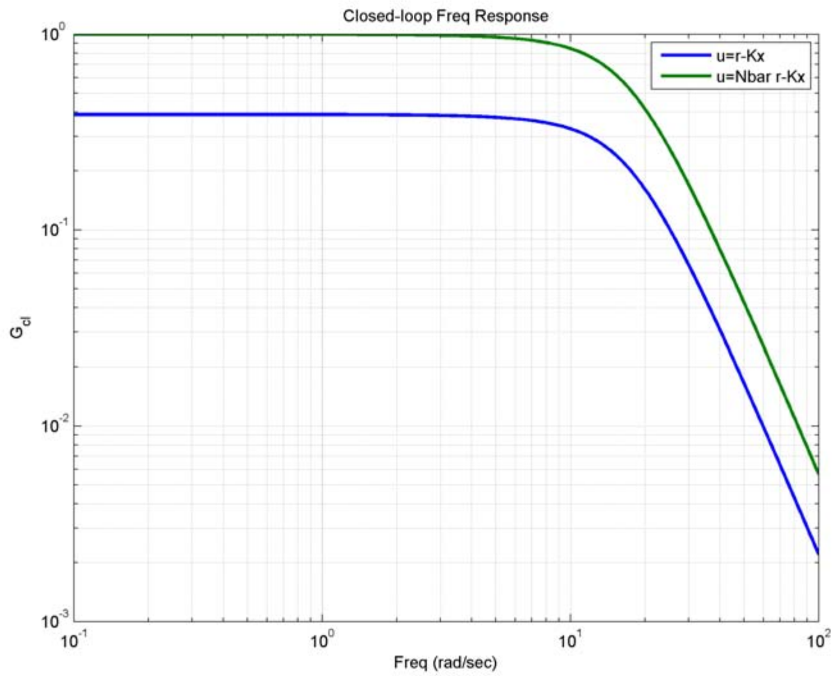


Fig. 7: Closed-loop frequency response. Clearly shows unity DC gain

- The worst possible. . . Unstable, NMP!!

$$G(s) = \frac{(s-1)}{(s+1)(s-3)}$$

  - Target pole locations $-1 \pm \mathbf{i}$

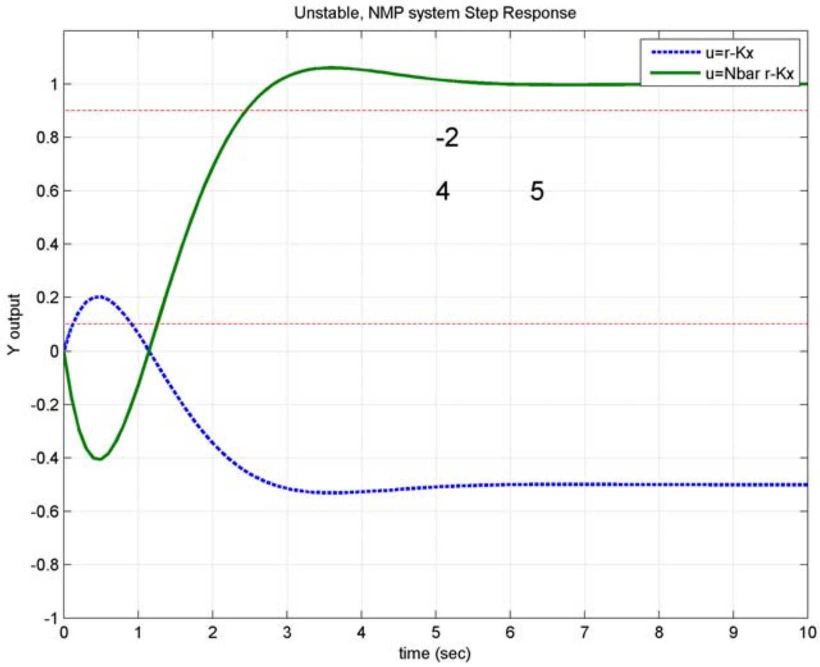


Fig. 8: Response to step input with and without the $\overline{N}$ correction. Gives the desired steady-state behavior, with little difficulty!

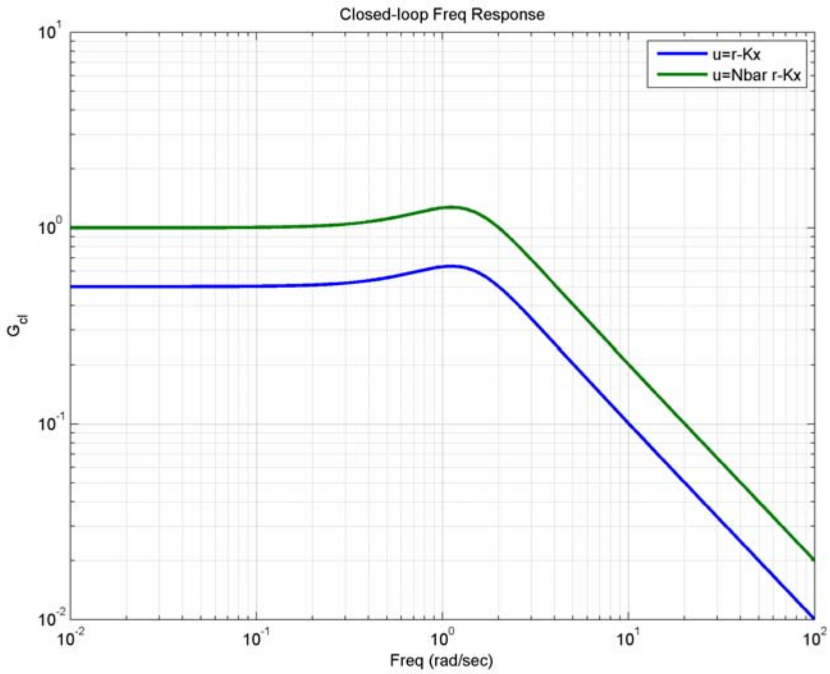


Fig. 9: Closed-loop frequency response. Clearly shows unity DC gain

FSFB Summary

- Full state feedback process is quite simple as it can be automated in Matlab using `acker` and/or `place`

- With more than 1 actuator, we have more than n degrees of freedom in the control $\rightarrow$ we can change the eigenvectors as desired, as well as the poles.

- The real issue now is where to put the poles...

- And to correct the fact that we cannot usually measure the state $\rightarrow$ develop an estimator.

Code: Step Response (step3.m)

```matlab
% Examples of pole placement with FSFB
% demonstrating the Nbar modifcation to the reference command
%
% Jonathan How
% Sept, 2010
%
close all;clear all
set(0,'DefaultLineLineWidth',2)
set(0,'DefaultlineMarkerSize',10);set(0,'DefaultlineMarkerFace','b')
set(0, 'DefaultAxesFontSize', 14);set(0, 'DefaultTextFontSize', 14);

% system
[a,b,c,d]=tf2ss(8*14*20,conv([1 8],conv([1 14],[1 20])));
% controller gains to place poles at specified locations
k=place(a,b,[-12+12*j;-12-12*j;-20]);

% find the feedforward gains
Nbar=-inv(c*inv(a-b*k)*b);

sys1=ss(a-b*k,b,c,d);
sys2=ss(a-b*k,b*Nbar,c,d);

t=[0:.01:1];
[y,t,x]=step(sys1,t);
[y2,t2,x2]=step(sys2,t);

figure(1);clf
plot(t,y,'--',t2,y2,'LineWidth',2);axis([0 1 0 1.2]);grid;
legend('u=r-Kx','u=Nbar r-Kx');xlabel('time (sec)');ylabel('Y output')
title('Step Response')
hold on
plot(t2([1 end]),[.1 .1]*y2(end),'r--');
plot(t2([1 end]),[.1 .1]*9*y2(end),'r--');
hold off

text(.4,.6,['k= [ ',num2str(round(k*1000)/1000),' ]'],'FontSize',14)
text(.4,.8,['Nbar= ',num2str(round(Nbar*1000)/1000)],'FontSize',14)
export_fig triple1 -pdf

figure(1);clf
f=logspace(-1,2,400);
gcl1=freqresp(sys1,f);
gcl2=freqresp(sys2,f);
loglog(f,abs(squeeze(gcl1)),f,abs(squeeze(gcl2)),'LineWidth',2);grid
xlabel('Freq (rad/sec)')
ylabel('G_{cl}')
title('Closed-loop Freq Response')
legend('u=r-Kx','u=Nbar r-Kx')
export_fig triple11 -pdf

%%%%%%%%%
% example 2
%
clear all

[a,b,c,d]=tf2ss(8*14*20,conv([1 -8],conv([1 -14],[1 -20])))
k=place(a,b,[-12+12*j;-12-12*j;-20])
% find the feedforward gains
Nbar=-inv(c*inv(a-b*k)*b);

sys1=ss(a-b*k,b,c,d);
sys2=ss(a-b*k,b*Nbar,c,d);

t=[0:.01:1];
[y,t,x]=step(sys1,t);
[y2,t2,x2]=step(sys2,t);

figure(2);clf
plot(t,y,'--',t2,y2,'LineWidth',2);axis([0 1 0 1.2])
grid;
legend('u=r-Kx','u=Nbar r-Kx')
xlabel('time (sec)');ylabel('Y output');title('Step Response')
hold on
plot(t2([1 end]),[.1 .1]*y2(end),'r--');
```

```
plot(t2([1 end]),[.1 .1]*9*y2(end),'r--');
hold off

text(.4,.6,['k= [ ',num2str(round(k*1000)/1000),' ]'],'FontSize',14)
text(.4,.8,['Nbar= ',num2str(round(Nbar*1000)/1000)],'FontSize',14)
export_fig triple2 -pdf

figure(2);clf
f=logspace(-1,2,400);
gcl1=freqresp(sys1,f);
gcl2=freqresp(sys2,f);
loglog(f,abs(squeeze(gcl1)),f,abs(squeeze(gcl2)),'LineWidth',2);grid
xlabel('Freq (rad/sec)')
ylabel('G_{cl}')
title('Closed-loop Freq Response')
legend('u=r-Kx','u=Nbar r-Kx')
export_fig triple21 -pdf

%%%%%%%%%%%%%%
% example 3
clear all

[a,b,c,d]=tf2ss(.94,[1 0 -0.0297])
k=place(a,b,[-1+j;-1-j]/4)
% find the feedforward gains
Nbar=-inv(c*inv(a-b*k)*b);

sys1=ss(a-b*k,b,c,d);
sys2=ss(a-b*k,b*Nbar,c,d);

t=[0:.1:30];
[y,t,x]=step(sys1,t);
[y2,t2,x2]=step(sys2,t);

figure(3);clf
plot(t,y,'--',t2,y2,'LineWidth',2);axis([0 30 0 2])
grid;
legend('u=r-Kx','u=Nbar r-Kx')
xlabel('time (sec)');ylabel('Y output');title('Step Response')
hold on
plot(t2([1 end]),[.1 .1]*y2(end),'r--');
plot(t2([1 end]),[.1 .1]*9*y2(end),'r--');
hold off

text(15,.6,['k= [ ',num2str(round(k*1000)/1000),' ]'],'FontSize',14)
text(15,.8,['Nbar= ',num2str(round(Nbar*1000)/1000)],'FontSize',14)
export_fig triple3 -pdf

figure(3);clf
f=logspace(-3,1,400);
gcl1=freqresp(sys1,f);
gcl2=freqresp(sys2,f);
loglog(f,abs(squeeze(gcl1)),f,abs(squeeze(gcl2)),'LineWidth',2);grid
xlabel('Freq (rad/sec)')
ylabel('G_{cl}')
title('Closed-loop Freq Response')
legend('u=r-Kx','u=Nbar r-Kx')
export_fig triple31 -pdf

%%%%%%%%%%%%
% example 4
clear all

[a,b,c,d]=tf2ss([1 -1],conv([1 1],[1 -3]))
k=place(a,b,[[-1+j;-1-j]])
% find the feedforward gains
Nbar=-inv(c*inv(a-b*k)*b);

sys1=ss(a-b*k,b,c,d);
sys2=ss(a-b*k,b*Nbar,c,d);

t=[0:.1:10];
[y,t,x]=step(sys1,t);
[y2,t2,x2]=step(sys2,t);

figure(3);clf
plot(t,y,'--',t2,y2,'LineWidth',2);axis([0 10 -1 1.2])
```

```
grid;
legend('u=r-Kx','u=Nbar r-Kx')
xlabel('time (sec)');ylabel('Y output')
title('Unstable, NMP system Step Response')
hold on
plot(t2([1 end]),[.1 .1]*y2(end),'r--');
plot(t2([1 end]),[.1 .1]*9*y2(end),'r--');
hold off

text(5,.6,['k= [ ',num2str(round(k*1000)/1000),' ]'],'FontSize',14)
text(5,.8,['Nbar= ',num2str(round(Nbar*1000)/1000)],'FontSize',14)
export_fig triple4 -pdf

figure(4);clf
f=logspace(-2,2,400);
gcl1=freqresp(sys1,f);
gcl2=freqresp(sys2,f);
loglog(f,abs(squeeze(gcl1)),f,abs(squeeze(gcl2)),'LineWidth',2);grid
xlabel('Freq (rad/sec)')
ylabel('G_{cl}')
title('Closed-loop Freq Response')
legend('u=r-Kx','u=Nbar r-Kx')
export_fig triple41 -pdf
```

16.30 / 16.31 Feedback Control Systems
Fall 2010