MECH 6323 - Robust Control - Midterm Exam

Author: Jonas Wagner

Date: 2022-04-02

```
clear
close all
```

Problem 4 - Plane Autopilot System

Controller Definition

Parameters

```
K_a = -1.5e-3;
K_q = -0.32;
a_z = 2;
a_q = 6;
```

State Matrices

```
A_C = [ 0           0
        K_q*a_q   0];
B_C = [ K_a*a_z           0
        K_a*K_q*a_q     K_q*a_q];
C_C = [ K_q    1];
D_C = [ K_a*K_q       K_q];
```

System Definition

```
sys_C = ss(A_C, B_C, C_C, D_C)
```

```
sys_C =

  A =
           x1     x2
   x1       0      0
   x2   -1.92      0

  B =
             u1       u2
   x1    -0.003        0
   x2   0.00288    -1.92

  C =
           x1     x2
   y1   -0.32      1

  D =
             u1       u2
   y1   0.00048    -0.32

Continuous-time state-space model.
```

```
tf_C = tf(sys_C)
```

```
tf_C =

  From input 1 to output:
  0.00048 s^2 + 0.00384 s + 0.00576
  ---------------------------------
                 s^2

  From input 2 to output:
  -0.32 s - 1.92
  --------------
        s

Continuous-time transfer function.
```

Plant Definition

Parameters

```
V = 886.78;
zeta = 0.6;
omega = 113;
```

Uncertain Coeficients

Nominal Values

```
Z_alpha_0 = -1.3046;
Z_delta_0 = -0.2142;
M_alpha_0 = 47.7109;
M_delta_0 = -104.83436;
```

Uncertain Dynamics

```
Z_alpha = @(k, delta) Z_alpha_0 * (1 + k * delta);
Z_delta = @(k, delta) Z_delta_0 * (1 + k * delta);
M_alpha = @(k, delta) M_alpha_0 * (1 + k * delta);
M_delta = @(k, delta) M_delta_0 * (1 + k * delta);
```

System Matrices

```
A_P = @(Z_alpha, Z_delta, M_alpha, M_delta) [
    Z_alpha 1 Z_delta     0
    M_alpha 0 M_delta     0
    0     0 0         1
    0     0 -omega^2 -2*zeta*omega
];
B_P = [
    0
    0
    0
    omega^2
];
```

```
C_P = @(Z_alpha, Z_delta) [
    V*Z_alpha   0   V*Z_delta   0
    0           1   0           0
];
D_P = [
    0
    0
];
```

Nominal System

Nominal System Matrices

```
A_P_0 = A_P(Z_alpha_0, Z_delta_0, M_alpha_0, M_delta_0)
```

```
A_P_0 = 4×4
10^4 ×
   -0.0001    0.0001   -0.0000         0
    0.0048         0   -0.0105         0
         0         0         0    0.0001
         0         0   -1.2769   -0.0136
```

```
% double(vpa(subs(A_P, ...
%     [Z_alpha, Z_delta, M_alpha, M_delta], ...
%     [Z_alpha_0, Z_delta_0, M_alpha_0, M_delta_0]), 4));
B_P_0 = B_P
```

```
B_P_0 = 4×1
           0
           0
           0
       12769
```

```
% double(vpa(subs(B_P, ...
%     [Z_alpha, Z_delta, M_alpha, M_delta], ...
%     [Z_alpha_0, Z_delta_0, M_alpha_0, M_delta_0]), 4));
C_P_0 = C_P(Z_alpha_0, Z_delta_0)
```

```
C_P_0 = 2×4
10^3 ×
   -1.1569         0   -0.1899         0
         0    0.0010         0         0
```

```
% double(vpa(subs(C_P, ...
%     [Z_alpha, Z_delta, M_alpha, M_delta], ...
%     [Z_alpha_0, Z_delta_0, M_alpha_0, M_delta_0]), 4));
D_P_0 = D_P
```

```
D_P_0 = 2×1
     0
     0
```

```
% double(vpa(subs(D_P, ...
%     [Z_alpha, Z_delta, M_alpha, M_delta], ...
%     [Z_alpha_0, Z_delta_0, M_alpha_0, M_delta_0]), 4));
```

Nominal State-space System

```
sys_P_0 = ss(A_P_0, B_P_0, C_P_0, D_P_0)
```

```
sys_P_0 =

  A = 
                x1          x2          x3          x4
   x1       -1.305           1     -0.2142           0
   x2        47.71           0      -104.8           0
   x3            0           0           0           1
   x4            0           0  -1.277e+04      -135.6

  B = 
               u1
   x1           0
   x2           0
   x3           0
   x4  1.277e+04

  C = 
            x1      x2      x3      x4
   y1    -1157       0  -189.9       0
   y2        0       1       0       0

  D = 
       u1
   y1   0
   y2   0

Continuous-time state-space model.
```

Nominal Transfer Function

```
tf_P_0 = tf(sys_P_0)
```

```
tf_P_0 =

  From input to output...
              -2.425e06 s^2 + 2.585e-08 s + 1.664e09
   1:  ----------------------------------------------------
       s^4 + 136.9 s^3 + 1.29e04 s^2 + 1.019e04 s - 6.092e05

                      -1.339e06 s - 1.877e06
   2:  ----------------------------------------------------
       s^4 + 136.9 s^3 + 1.29e04 s^2 + 1.019e04 s - 6.092e05

Continuous-time transfer function.
```

Uncertain System Dynamics

Uncertain Matrices

```
A_P = @(k, Delta) A_P(  Z_alpha(k, Delta(1)), ...
                        Z_delta(k, Delta(2)), ...
                        M_alpha(k, Delta(3)), ...
                        M_delta(k, Delta(4)));
C_P = @(k, Delta) C_P(  Z_alpha(k, Delta(1)), ...
                        Z_delta(k, Delta(2)));
```

Uncertain System

```
sys_P = @(k, Delta) ss(A_P(k, Delta), B_P, C_P(k, Delta), D_P);
```

Feedback System Definition

Open Loop System: $L(s) = C(s)P(s)$

Nominal

```
sys_L_0 = series(sys_C, sys_P_0)
```

```
sys_L_0 =

  A = 
               x1          x2          x3          x4          x5          x6
   x1      -1.305           1     -0.2142           0           0           0
   x2       47.71           0      -104.8           0           0           0
   x3           0           0           0           1           0           0
   x4           0           0  -1.277e+04      -135.6       -4086   1.277e+04
   x5           0           0           0           0           0           0
   x6           0           0           0           0       -1.92           0

  B = 
            u1       u2
   x1        0        0
   x2        0        0
   x3        0        0
   x4    6.129    -4086
   x5   -0.003        0
   x6  0.00288    -1.92

  C = 
           x1      x2      x3      x4      x5      x6
   y1   -1157       0  -189.9       0       0       0
   y2       0       1       0       0       0       0

  D = 
       u1  u2
   y1   0   0
   y2   0   0

Continuous-time state-space model.
```

Uncertain

```
sys_L = @(k, Delta) series(sys_C, sys_P(k, Delta));
```

Closed Loop System: $S(s) = \dfrac{C(s)P(s)}{1 + C(s)P(s)}$

Nominal

```
sys_S_0 = feedback(sys_L_0, eye(size(C_P_0,1)))
```

```
sys_S_0 =

  A = 
              x1          x2          x3          x4          x5          x6
   x1     -1.305           1     -0.2142           0           0           0
   x2      47.71           0      -104.8           0           0           0
```

```
 x3         0         0         0         1         0         0
 x4      7091      4086  -1.16e+04    -135.6     -4086  1.277e+04
 x5    -3.471         0   -0.5698         0         0         0
 x6     3.332      1.92    0.5471         0     -1.92         0

B =
          u1       u2
 x1        0        0
 x2        0        0
 x3        0        0
 x4    6.129    -4086
 x5   -0.003        0
 x6  0.00288    -1.92

C =
         x1      x2      x3      x4      x5      x6
 y1   -1157       0  -189.9       0       0       0
 y2       0       1       0       0       0       0

D =
      u1  u2
 y1    0   0
 y2    0   0

Continuous-time state-space model.
```

Uncertain

```
sys_S = @(k, Delta) feedback(sys_L(k,Delta), eye(size(C_P_0,1)));
```

(a)

```
eig_S_0 = eig(sys_S_0)
```

```
eig_S_0 = 6×1 complex
 -39.2033 +71.5951i
 -39.2033 -71.5951i
 -49.2597 + 0.0000i
  -1.8309 + 0.0000i
  -3.7038 + 1.4953i
  -3.7038 - 1.4953i
```

Since $\mathscr{R}(\lambda_i) < 0 \ \forall_i$, the closed loop system $S(s)$ is stable.

```
sys_S_stable = isstable(sys_S_0)
```

```
sys_S_stable = logical
   1
```

(b)

Random Testing Code

```
i_worst = 1;
k_try = 10;
N_samples = 10000;
Delta_data = 2 * rand([4, N_samples]) - 1;
```

```
for i = 1:N_samples
    % Better Implimentation
    sys_S_test = @(k) sys_S(k, Delta_data(:,i));
    if isstable(sys_S_test(k_try))
        break
    else
        while ~isstable(sys_S_test(k_try))
            k_try = 0.99 * k_try;
            i_worst = i;
        end
    end
    % Alternative (as described in the problem itself)
%     Delta = Delta_data(:,i);
%     sys_S_unstable = true;
%     while sys_S_unstable
%         sys_S_test = sys_S(k_try, Delta);
%         if max(real(eig(sys_S_test))) >= 0
%             k_try = 0.99 * k_try;
%             i_worst = i;
%         else
%             sys_S_unstable = false;
%         end
%     end
end
```

Random Testing Results

```
k_bar = k_try
```

```
k_bar = 1.1639
```

```
Delta_worst = Delta_data(:,i)
```

```
Delta_worst = 4×1
    0.9599
    0.9308
   -0.6538
   -0.5855
```

Since this gets randomized everytime, $\bar{k}$ changes but often gets down below 1 to 0.7-ish, but also stays at 10 sometimes.

Why must $k_{max} \leq \bar{k}$?

Well, many reasons. We know that $k_{max}$is the largest possible k that maintains stability of the closed-loop system, and therefore all unstabilizing k due to some disturbance would be greater then the lowest-upper bound on k, i.e. $\bar{k} \geq \sup_k S(s)$ stable. Therefore, $k_{max} = \sup_k S(s) \leq \bar{k}$.

(c)

Uncertain System

Bounded Uncertainty

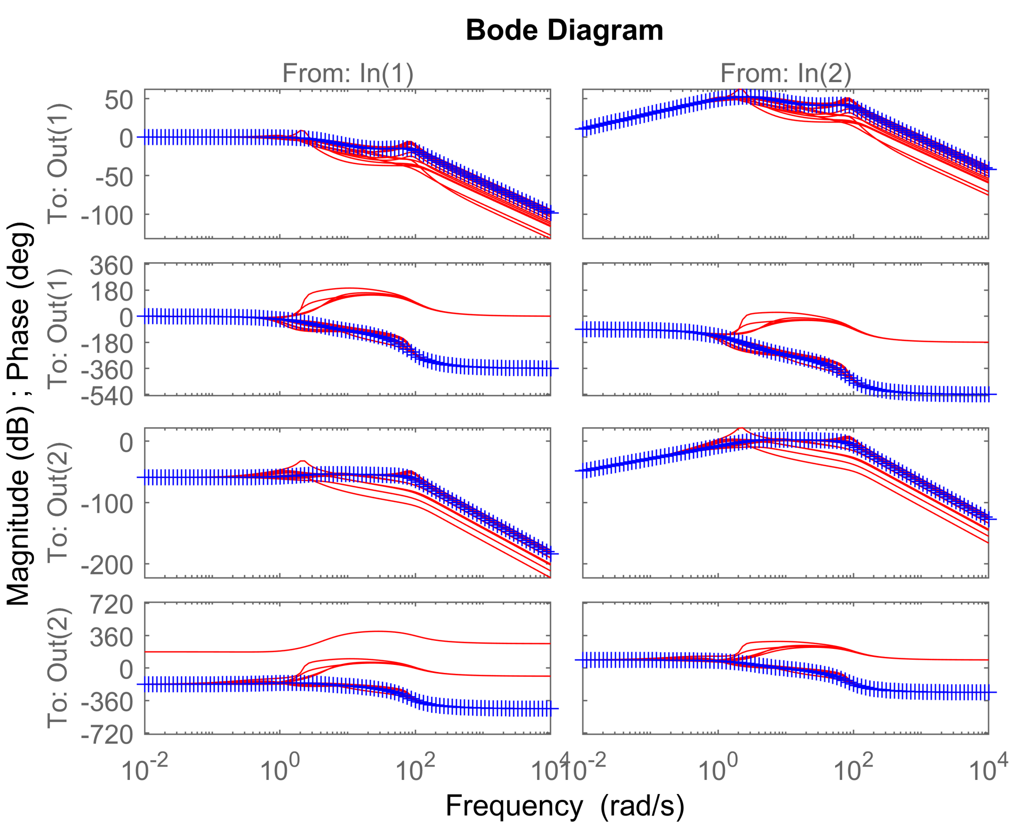
```
delta_1_u = ureal('delta_1', 0);
delta_2_u = ureal('delta_2', 0);
delta_3_u = ureal('delta_3', 0);
delta_4_u = ureal('delta_4', 0);

Delta_u = [
    delta_1_u
    delta_2_u
    delta_3_u
    delta_4_u
];
```

Uncertain SS system

```
sys_S_u = @(k) sys_S(k, Delta_u);
```

Uncertain System Frequency Respons

```
w_min = -2;
w_max = 4;
freqs = logspace(w_min,w_max,100);
sys_S_u_k1 = sys_S_u(1);
usysfrd = ufrd(sys_S_u_k1, freqs);
bode(usysfrd,'r', usysfrd.NominalValue, 'b+')
```



robostab

```
opts = robOptions(  'Display','on', ...
                    'VaryFrequency','on',...
                    'Sensitivity','on');
[stabmarg, destabunc, report] = robstab(sys_S_u_k1, opts)
```

```
Computing bounds... Points completed: 42/42
Computing peak...  Percent completed: 100/100
System is not robustly stable for the modeled uncertainty.
 -- It can tolerate up to 60.3% of the modeled uncertainty.
 -- There is a destabilizing perturbation amounting to 60.4% of the modeled uncertainty.
 -- This perturbation causes an instability at the frequency 3.44 rad/seconds.
 -- Sensitivity with respect to each uncertain element is:
      3% for delta_1. Increasing delta_1 by 25% decreases the margin by 0.75%.
      1% for delta_2. Increasing delta_2 by 25% decreases the margin by 0.25%.
      16% for delta_3. Increasing delta_3 by 25% decreases the margin by 4%.
      42% for delta_4. Increasing delta_4 by 25% decreases the margin by 10.5%.
stabmarg = struct with fields:
           LowerBound: 0.6035
           UpperBound: 0.6045
    CriticalFrequency: 3.4386
destabunc = struct with fields:
    delta_1: -0.6045
    delta_2: -0.6045
    delta_3: 0.6045
    delta_4: -0.6045
report = struct with fields:
                Model: 1
            Frequency: [44×1 double]
               Bounds: [44×2 double]
    WorstPerturbation: [44×1 struct]
          Sensitivity: [1×1 struct]
```

From this we have $k_{max} \approx 0.6045$ to result in $k * \Delta = 0.6045 * [-1 \quad -1 \quad 1 \quad -1]^T$.

This value is consistant with the numerical results as it is below, but not too small by comparrision to what is expected.

Additionally, we know that the critical frequency that this is occurring at is

----

(d)

```
k_max = 0.6045;
sys_S_critical = sys_S(k_max, [-1; -1; 1; -1])
```

```
sys_S_critical =

  A = 
             x1          x2          x3          x4          x5          x6
   x1    -0.516           1    -0.08472           0           0           0
   x2     76.55           0      -41.46           0           0           0
   x3         0           0           0           1           0           0
   x4      2804        4086  -1.231e+04      -135.6       -4086   1.277e+04
   x5    -1.373           0     -0.2254           0           0           0
   x6     1.318        1.92      0.2164           0       -1.92           0
```

```
  B = 
            u1       u2
   x1        0        0
   x2        0        0
   x3        0        0
   x4    6.129    -4086
   x5   -0.003        0
   x6  0.00288    -1.92
 
  C = 
           x1      x2      x3      x4      x5      x6
   y1  -457.6       0  -75.12       0       0       0
   y2       0       1       0       0       0       0
 
  D = 
       u1  u2
   y1   0   0
   y2   0   0
 
Continuous-time state-space model.
```

```
tf_S_critical = tf(sys_S_critical)
```

```
tf_S_critical =
 
  From input 1 to output...
                   -460.4 s^4 - 3684 s^3 + 1.46e05 s^2 + 1.212e06 s + 1.818e06
   1:  ---------------------------------------------------------------------------------
       s^6 + 136.1 s^5 + 1.23e04 s^4 + 1.619e05 s^3 + 2.989e05 s^2 + 1.896e06 s + 1.818e06
 
                              -254.1 s^3 - 2204 s^2 - 4416 s - 2050
   2:  ---------------------------------------------------------------------------------
       s^6 + 136.1 s^5 + 1.23e04 s^4 + 1.619e05 s^3 + 2.989e05 s^2 + 1.896e06 s + 1.818e06
 
  From input 2 to output...
                      3.07e05 s^4 + 1.842e06 s^3 - 1.01e08 s^2 - 6.061e08 s
   1:  ---------------------------------------------------------------------------------
       s^6 + 136.1 s^5 + 1.23e04 s^4 + 1.619e05 s^3 + 2.989e05 s^2 + 1.896e06 s + 1.818e06
 
                       1.694e05 s^3 + 1.13e06 s^2 + 6.835e05 s - 5.387e-11
   2:  ---------------------------------------------------------------------------------
       s^6 + 136.1 s^5 + 1.23e04 s^4 + 1.619e05 s^3 + 2.989e05 s^2 + 1.896e06 s + 1.818e06
 
Continuous-time transfer function.
```

```
eig_S_critical = eig(sys_S_critical)
```

```
eig_S_critical = 6×1 complex
 -60.4635 +82.3580i
 -60.4635 -82.3580i
 -14.1488 + 0.0000i
   0.0006 + 3.4385i
   0.0006 - 3.4385i
  -1.0412 + 0.0000i
```

As can be seen by the eigenvalues at $\lambda_{4,5} = 0.00 \pm j3.44$, the poles of the system cross the $j\omega$-axis to become unstable at $k_{max} \approx 0.6045$ with $\omega \approx 3.44$. This confirms the robostab estimates to force the system to become unstable at the maximum perturbation of $\Delta = [-1 \quad -1 \;\; 1 \quad -1]^T$.