University of Colorado - Boulder

Aircraft Dynamics - Fall 2019

ASEN 3128
Lab Section 011

ASEN 3128 Assignment 11

Author:
Grant Norman - 107478513

Professor:
Professor Nisar Ahmed

December 5, 2019

College of Engineering & Applied Science
UNIVERSITY OF COLORADO BOULDER

Structure of Assignment

This assignment builds on assignment 10, so most of the code from that assignment is reused here. Further, the guiding equations are implemented in the MATLAB code, which is attached. The code is the best way to show the work for the questions, although brief explanations and the analysis are provided directly in this assignment.

Question 1

A. Spiral Mode Approximation

The spiral mode may be approximated by using the last two terms of the characteristic equation $d\lambda + e = 0$. Rearranging, we have that $\lambda \approx -e/d$. From Etkin Eq. 6.8,3-4, e and d are given in Eq. 1. Note that d is also approximated by ignoring smaller terms; this approximation is what is actually used as d. The script terms are entries of the $\mathbf{A_{lat}}$ matrix, which is included within questions 2 and 3 as the upper 4×4 square matrix in the $\mathbf{A_{lat,aug}}$ matrix.

$$\begin{aligned} e &= g[(\mathscr{L}_v\mathscr{N}_r - \mathscr{L}_r\mathscr{N}_v)\cos\theta_0 + (\mathscr{L}_p\mathscr{N}_v - \mathscr{L}_v\mathscr{N}_p)\sin\theta_0] \\ d &= -g(\mathscr{L}_v\cos\theta_0 + \mathscr{N}_v\sin\theta_0) + \mathscr{Y}_v(\mathscr{L}_r\mathscr{N}_p - \mathscr{L}_p\mathscr{N}_r) + \mathscr{Y}_r(\mathscr{L}_p\mathscr{N}_v - \mathscr{L}_v\mathscr{N}_p) \\ d &\approx -g(\mathscr{L}_v\cos\theta_0 + \mathscr{N}_v\sin\theta_0) + u_0(\mathscr{L}_v\mathscr{N}_p - \mathscr{L}_p\mathscr{N}_v) \end{aligned} \tag{1}$$

Similarly, Etkin Eq. 6.8,8 gives the roll mode approximation as $\lambda \approx \mathscr{L}_p$.

From assignment 10, we have the full matrix calculations of these eigenvalues, which are compared to the approximations. We find that $\lambda_{spiral} = -0.00510 \approx -0.00513$ and $\lambda_{roll} = -0.700 \approx -0.559$. Thus, the spiral mode approximation is more accurate in this case, with a negligible error ($< 1\%$) from the true value. The roll mode approximation is noticeably further (18%) from the true value. For these flight conditions, this was the case, although for a different trim state, the approximations may be closer or further from the true values. Nonetheless, including the full 4×4 matrix eigenvalue calculation will give precise results with about 4 times the processing required as for these approximation.

Question 2

Part a

The non-dimensional derivatives from Table 7.3 in Etkin were dimensionalized by the factors specified in Table 4.1 in Etkin. That is, the non-dimensional derivatives for Y were multiplied by $\rho \cdot u_0^2 \cdot S$, and the non-dimensional derivatives for L and N were multiplied by $\rho \cdot u_0^2 \cdot b \cdot S$. This gives the dimensional control derivatives found in Table 1, where the units are the units along the top, divided by the units along the side.

	Y (N)	L (Nm)	N (Nm)
δ_a (rad)	0	-3391451	-48913
δ_r (rad)	476000	1729441	-31162673

Table 1 Dimensional Control Derivatives

Then, the **B** matrix was found through Equation 7.9,3 in Etkin. This yields the matrix given in Eq. 2 below.

$$\mathbf{B} = \begin{bmatrix} 0 & 1.64973 \\ -0.13688 & 0.14008 \\ 0.00690 & -0.47227 \\ 0 & 0 \end{bmatrix} \tag{2}$$

Part b

Next, the augmented matrices are found. The $\mathbf{A_{lat,aug}}$ matrix corresponds to the same elements as the regular $\mathbf{A_{lat}}$ matrix in the upper 4×4 submatrix. The dynamics of the rest of the matrix is given by Equation 4.9,19 in Etkin.

Similarly, the $\mathbf{B_{lat,aug}}$ matrix has the same elements as the regular $\mathbf{B_{lat}}$ matrix in the upper 4×2 submatrix, with the other entries as zeros. These matrices are included below in Eq. 3 and 4, respectively.

$$\mathbf{A_{lat,aug}} = \begin{bmatrix} -0.07997 & 0 & -157.9 & 9.81 & 0 & 0 \\ -0.01695 & -0.5592 & 0.6126 & 0 & 0 & 0 \\ 0.00624 & 0.004773 & -0.00247 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 157.89 & 0 \end{bmatrix} \tag{3}$$

$$\mathbf{B_{lat,aug}} = \begin{bmatrix} 0 & 1.64973 \\ -0.13688 & 0.14008 \\ 0.00690 & -0.47227 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{4}$$

Question 3

In general, Eq. 5 specifies the approach for calculating the $[\mathbf{A_{lat,CL}}]_{6\times6}$ augmented, closed loop lateral dynamics matrix. This matrix then describes the dynamics of the system, given by Eq. 6. Thus, the eigenvalues of interest are of the $\mathbf{A_{lat,CL}}$ matrix. Note that the entries of the $[\mathbf{K}]$ matrix are positive for *negative* control and negative for *positive* control. Note that the sign of the variable k is always *positive*. Further, the eigenvalues at $\lambda = 0$ from the added states are omitted for clarity.

$$[\mathbf{A_{lat,CL}}]_{6\times6} = [\mathbf{A_{lat,aug}}]_{6\times6} - [\mathbf{B_{lat,aug}}]_{6\times2}[\mathbf{K}]_{2\times6} \tag{5}$$

$$\dot{\boldsymbol{y}}_{\boldsymbol{aug}} = [\mathbf{A_{lat,CL}}]\ \mathbf{y_{aug}} \tag{6}$$

$$\mathbf{y_{aug}} = \begin{bmatrix} \Delta v \\ \Delta q \\ \Delta r \\ \Delta \phi \\ \Delta \psi \\ \Delta y^E \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} \Delta \delta_a \\ \Delta \delta_r \end{bmatrix} \tag{7}$$

$$\mathbf{A_{lat,aug}} = \begin{bmatrix} \mathscr{Y}_v & \mathscr{Y}_p & \mathscr{Y}_r & g\cos\theta_0 & 0 & 0 \\ \mathscr{L}_v & \mathscr{L}_p & \mathscr{L}_r & 0 & 0 & 0 \\ \mathscr{N}_v & \mathscr{N}_p & \mathscr{N}_r & 0 & 0 & 0 \\ 0 & 1 & \tan\theta_0 & 0 & 0 & 0 \\ 0 & 0 & \sec\theta_0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & u_0\cos\theta_0 & 0\Delta\delta_r \end{bmatrix} \tag{8}$$

Part a

In this part, the bank angle $\Delta\phi$ is used with a positive feedback control of the aileron deflection $\Delta\delta_a$. Using Eq. 5 with the $\mathbf{K}$ matrix below, only column 4 of the $\mathbf{A_{lat,aug}}$ is modified. Thus, none of the script terms ($\mathscr{Y}$, $\mathscr{L}$, $\mathscr{N}$) in Eq. 8 are modified. Note that these script terms correspond to the stability derivatives of $\mathbf{A_{lat,CL}}$, so none of the stability derivatives of this matrix change either. This process of analysis is repeated for the subsequent parts.

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 & -k & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 1 shows the eigenvalue loci for the requested gain values. The roll mode becomes less stable as it moves towards the Imaginary axis. The spiral mode becomes more stable, moving in the negative Real direction. These two modes overlap on the real axis at the break-out point of about $\lambda = -0.34$, where these modes combine to form a new oscillatory mode, which then becomes less damped for higher gains. The dutch roll mode gains a greater natural frequency ω_n as its loci move farther from the origin. The loci also form a slightly smaller angle with the Real axis, meaning that the dutch roll mode is *very slightly* better damped (ζ slightly increases).

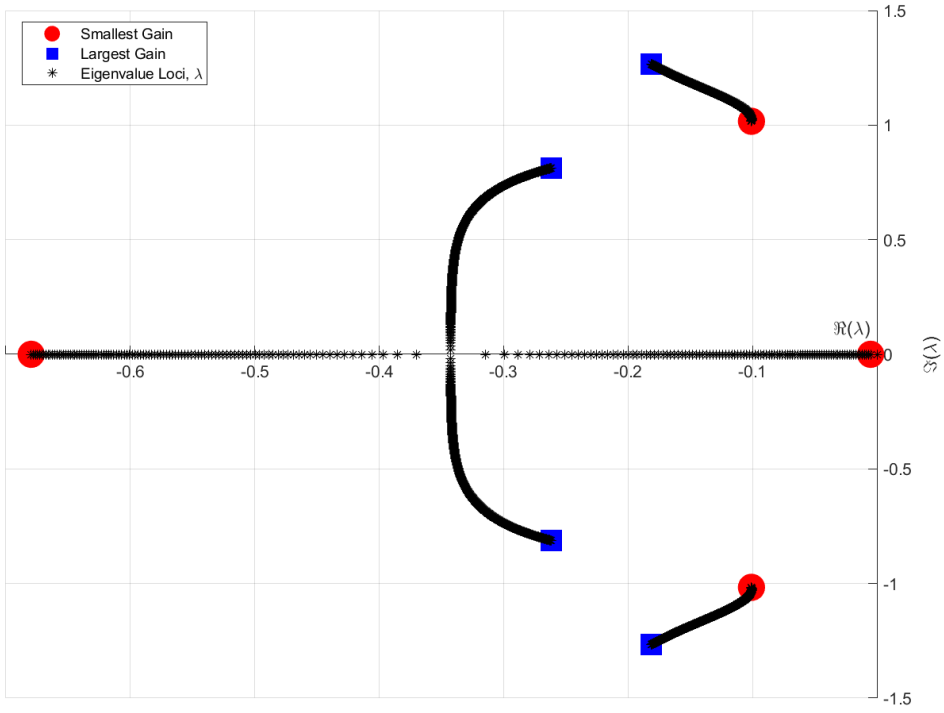


Fig. 1 Part a Eigenvalue Loci

Part b

In this part, the roll rate Δp is used with a *negative* feedback control of the aileron deflection $\Delta\delta_a$. Using Eq. 5 with the **K** matrix below, column 2 of the $\mathbf{A_{lat,aug}}$ is modified. As a result, the script terms $\mathscr{Y}_p$, $\mathscr{L}_p$, $\mathscr{N}_p$ in Eq. 8 are altered. These script terms correspond to the stability derivatives Y_p, L_p, and N_p of $\mathbf{A_{lat,CL}}$, which are similarly modified.

$$\mathbf{K} = \begin{bmatrix} 0 & k & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 2 shows the eigenvalue loci for the requested gain values. The roll mode becomes less stable as it moves towards the Imaginary axis. The spiral mode becomes more stable, moving in the negative Real direction. These two modes overlap on the real axis at the break-out point of about $\lambda = -0.061$, where these modes combine to form a new oscillatory mode, which then breaks in at $\lambda = 0.063$. This reestablishes the spiral and roll modes, with both of them being unstable, and going in different directions along the Real axis.

The dutch roll mode becomes slightly less damped, at first. Then, it is damped slightly, but still with less damping than no gains at all. The natural frequency generally decreases slightly.

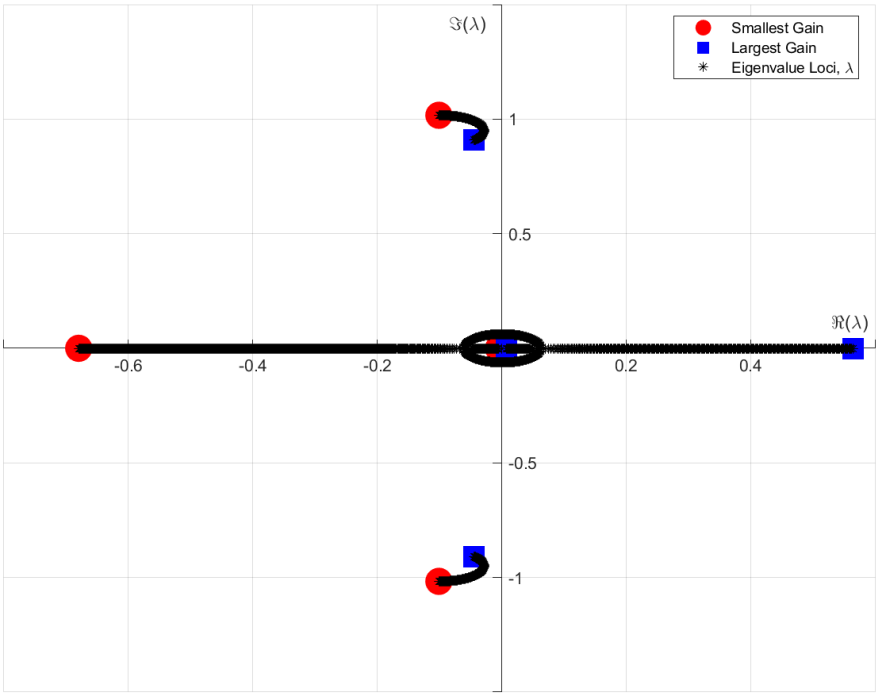


Fig. 2 Part b Eigenvalue Loci

Part c

In this part, the yaw rate Δp is used with a *negative* feedback control of the aileron deflection $\Delta\delta_a$. Using Eq. 5 with the **K** matrix below, column 3 of the $\mathbf{A_{lat,aug}}$ is modified. As a result, the script terms $\mathscr{Y}_r$, $\mathscr{L}_r$, $\mathscr{N}_r$ in Eq. 8 are altered. These script terms correspond to the stability derivatives Y_r, L_r, and N_r of $\mathbf{A_{lat,CL}}$, which are similarly modified.

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & k & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 3 shows the eigenvalue loci for the requested gain values. With increasing k values, the roll mode becomes further stabilized (more negative λ). The spiral mode is destabilized, with *lambda* moving from the negative real axis over to the positive real axis. Finally, the dutch roll mode is damped slightly, as the angle with the Real axis is decreased ($\cos\beta = \zeta$, and the natural frequency slightly increases.

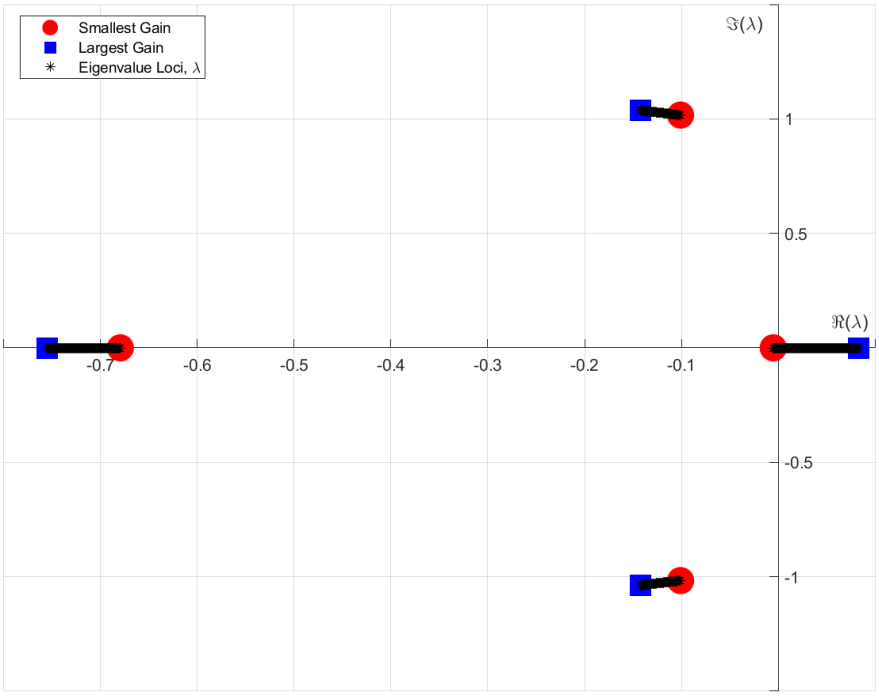


Fig. 3 Part c Eigenvalue Loci

Part d

In this part, the azimuth angle deviation $\Delta\psi$ is used with a *positive* feedback control of the aileron deflection $\Delta\delta_a$. Using Eq. 5 with the **K** matrix below, column 5 of the $\mathbf{A_{lat,aug}}$ is modified. As a result, all the script terms in Eq. 8 are unchanged. Similarly, all of the associated stability derivatives are unchanged.

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 & 0 & -k & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 4 shows the eigenvalue loci for the requested gain values. With increasing k values, the roll mode becomes further stabilized (more negative λ) - giving a smaller time constant τ. The spiral mode breaks out into an unstable oscillatory mode. Finally, the dutch roll mode is damped slightly more, as the angle with the Real axis is decreased ($\cos\beta = \zeta$, and the natural frequency slightly decreases, and then slightly increases.

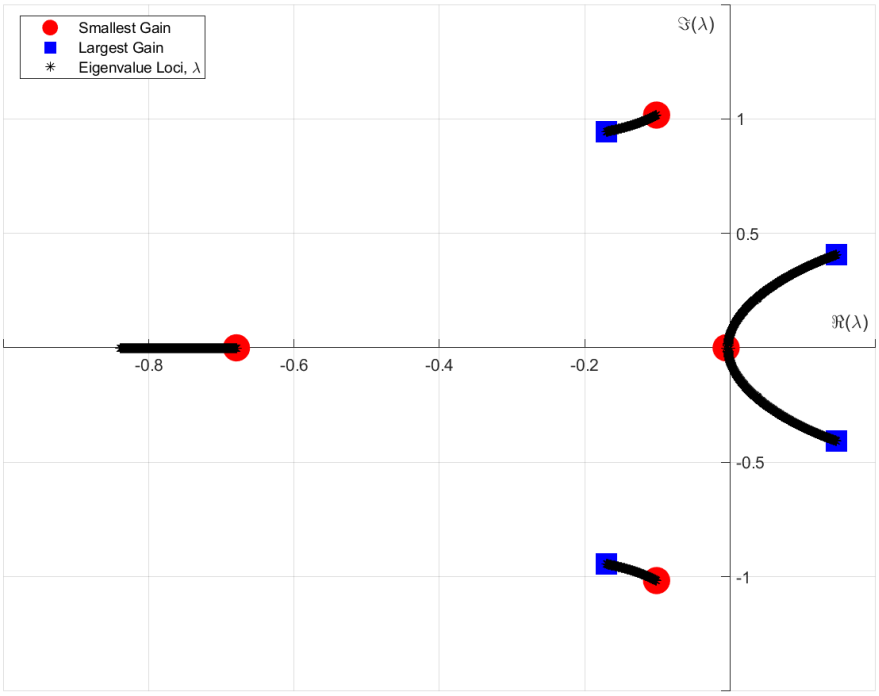


Fig. 4 Part d Eigenvalue Loci

Part e

In this part, the side velocity deviation Δv is used with a *negative* feedback control of the *rudder* deflection $\Delta\delta_r$. Using Eq. 5 with the **K** matrix below, column 1 of the $\mathbf{A_{lat,aug}}$ is modified. As a result, the script terms $\mathscr{Y}_v$, $\mathscr{L}_v$, $\mathscr{N}_v$ in Eq. 8 are altered.

Similarly, the stability derivatives Y_v, L_v, and N_v are effectively modified.

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ k & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 5 shows the eigenvalue loci for the requested gain values. With increasing k values, the roll mode moves in the positive direction along the real axis. The spiral mode becomes unstable and then breaks out into an oscillatory mode. Finally, the dutch roll mode becomes slightly less damped, and its natural frequency *significantly* increases.

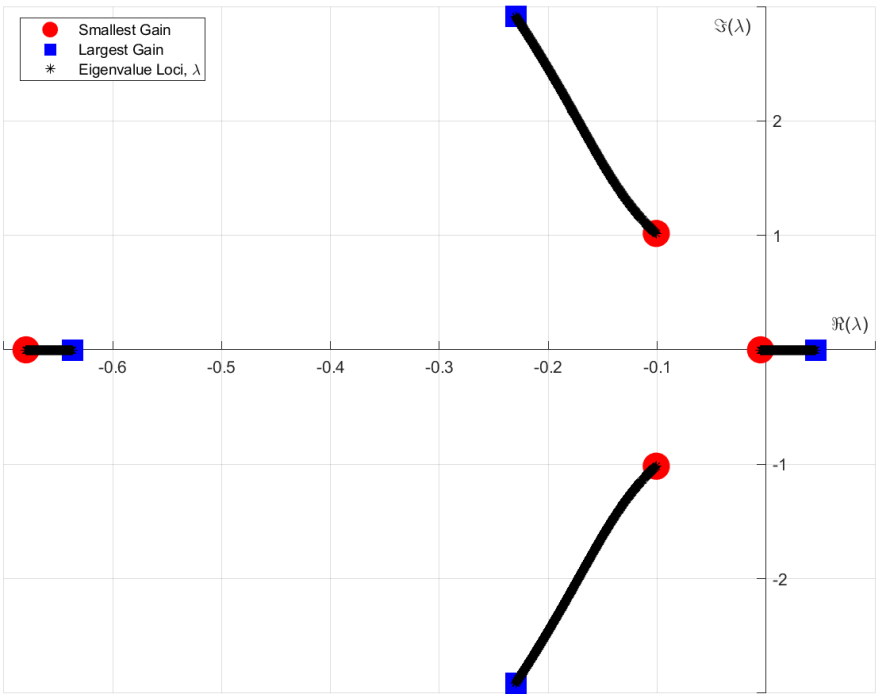


Fig. 5 Part e Eigenvalue Loci

Part f

In this part, the roll rate deviation Δp is used with a *negative* feedback control of the *rudder* deflection $\Delta\delta_r$. Using Eq. 5 with the **K** matrix below, column 2 of the $\mathbf{A}_{\mathbf{lat,aug}}$ is modified. As a result, the script terms $\mathscr{Y}_p$, $\mathscr{L}_p$, $\mathscr{N}_p$ in Eq. 8 are altered.

Similarly, the stability derivatives Y_p, L_p, and N_p are effectively modified.

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & k & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 6 shows the eigenvalue loci for the requested gain values. With increasing k values, the roll mode moves in the positive direction along the real axis, combining with the spiral mode to form an oscillatory mode. This mode then breaks in, and the roll and spiral modes remain unstable. Finally, the dutch roll mode becomes significantly more damped, and its natural frequency increases.

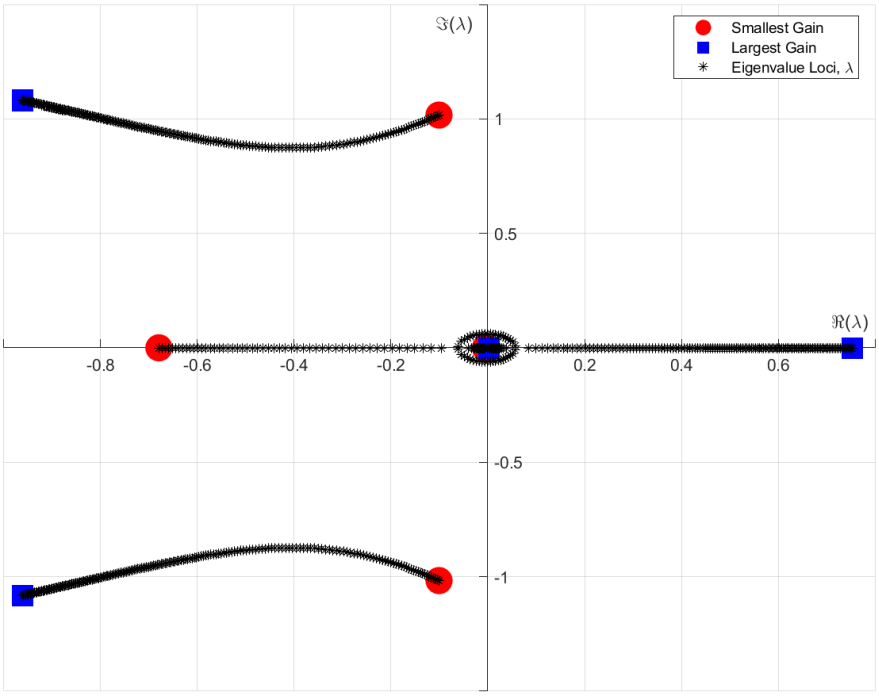


Fig. 6 Part f Eigenvalue Loci

Part g

In this part, the yaw rate deviation Δr is used with both *positive and negative* feedback for control of the *rudder* deflection $\Delta\delta_r$. Using Eq. 5 with the **K** matrix below, column 3 of the $\mathbf{A_{lat,aug}}$ is modified. The negative k corresponds to the positive gains, and the positive k corresponds to the negative feedback. As a result, the script terms $\mathscr{Y}_p$, $\mathscr{L}_r$, $\mathscr{N}_r$ in Eq. 8 are altered.

Similarly, the stability derivatives Y_r, L_r, and N_r are effectively modified.

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mp k & 0 & 0 & 0 \end{bmatrix}$$

Positive Feedback: -k

Figure 7 shows the eigenvalue loci for the requested gain values for *positive feedback*. As k increases, the dutch roll mode becomes much better damped, eventually reaching the Real axis and forming two non-oscillatory modes, going in either direction along the Real axis. Again, the roll mode and spiral mode move towards one another, and break out into an oscillatory mode. This oscillatory mode becomes less damped, and potentially unstable for higher gains.

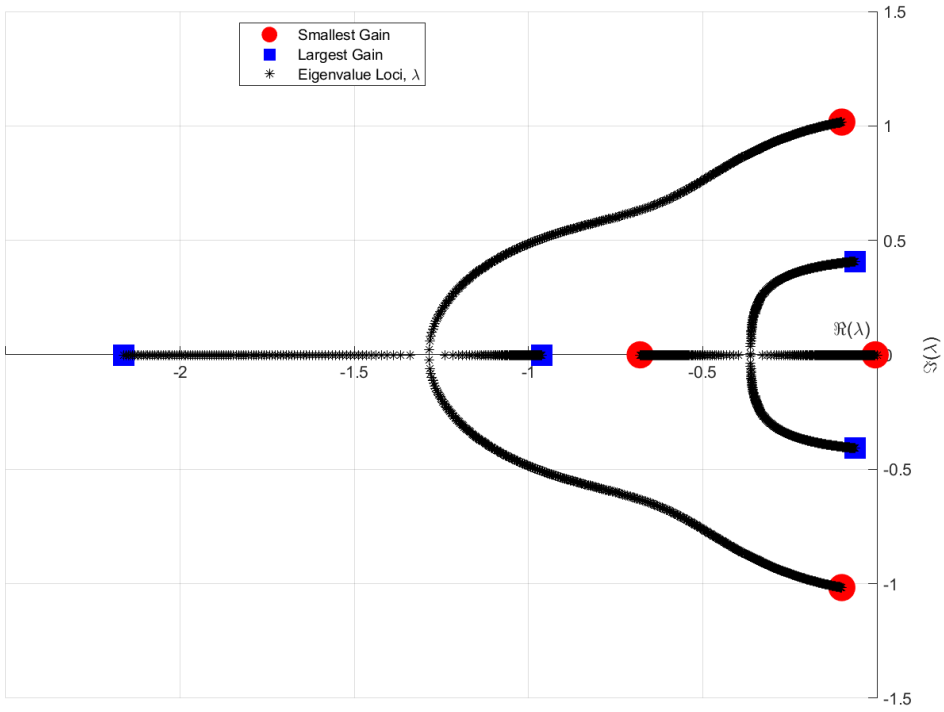


Fig. 7 Part g Positive Feedback Eigenvalue Loci

Negative Feedback: +k

Figure 7 shows the eigenvalue loci for the requested gain values for *negative feedback*. As k increases, the dutch roll mode becomes less damped and eventually unstable. The roll mode becomes more stable, with a smaller time constant. The spiral mode becomes less stable, moving along the positive real axis.

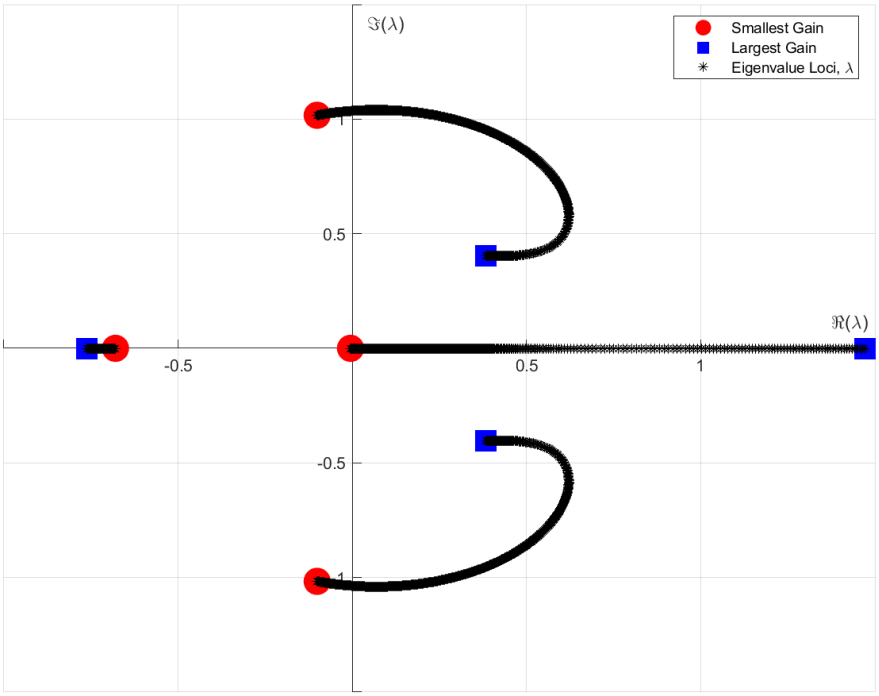


Fig. 8 Part g Negative Feedback Eigenvalue Loci

Part h

In this part, the bank angle deviation $\Delta\phi$ is used with both *negative and positive* feedback for control of the *rudder* deflection $\Delta\delta_r$. Using Eq. 5 with the $\mathbf{K}$ matrix below, column 4 of the $\mathbf{A}_{\mathbf{lat,aug}}$ is modified. The negative k corresponds to the positive gains, and the positive k corresponds to the negative feedback. We see that no script terms are modified, and thus no stability derivatives are augmented.

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \pm k & 0 & 0 \end{bmatrix}$$

Negative Feedback: $+k$

Figure 10 shows the eigenvalue loci for the requested gain values for *negative feedback*. As k increases, the dutch roll mode's natural frequency increases, and the damping slightly increases. The roll mode becomes more constant, with a smaller time constant. On the other hand, the spiral mode becomes unstable, moving in the positive real direction.

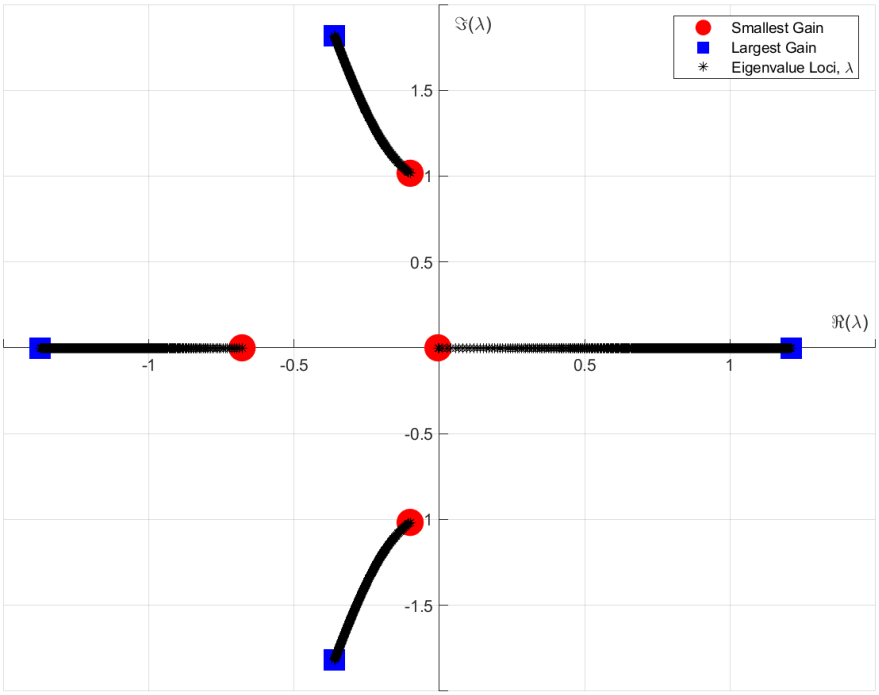


Fig. 9 Part h Negative Feedback Eigenvalue Loci

Positive Feedback: -k

Figure 10 shows the eigenvalue loci for the requested gain values for *positive feedback*. As k increases, the dutch roll mode becomes quickly unstable, although gaining more damping for a few gain values before the instability. The roll and spiral modes move towards one another on the negative real axis and break out into an oscillatory mode. This oscillatory mode then becomes less damped, with a higher natural frequency.

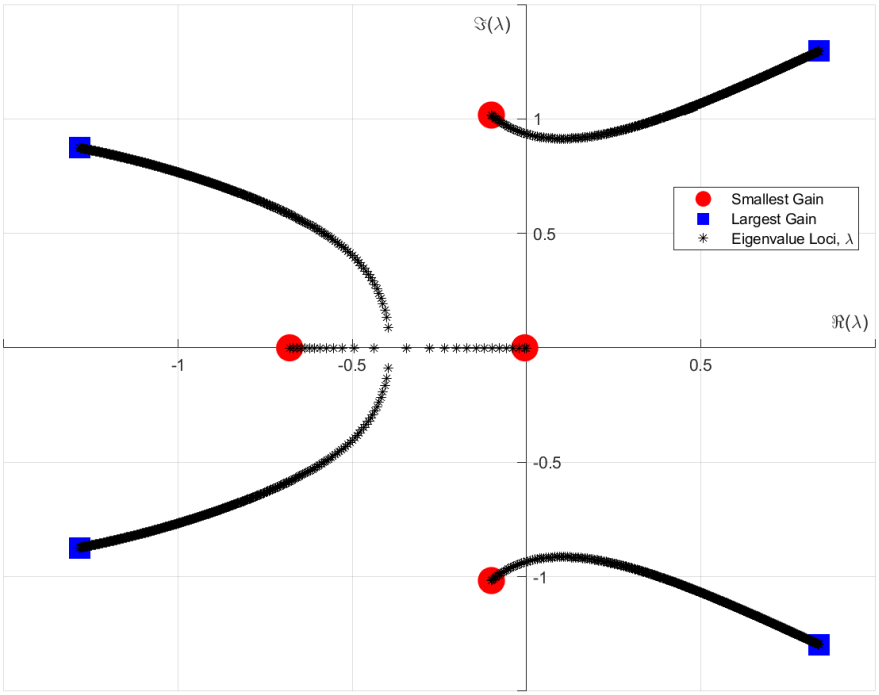


Fig. 10 Part h Positive Feedback Eigenvalue Loci

Part i

In this part, the azimuth deviation $\Delta\psi$ is used with both *positive and negative* feedback for control of the *rudder* deflection $\Delta\delta_r$. Using Eq. 5 with the **K** matrix below, column 5 of the $\mathbf{A_{lat,aug}}$ is modified. The negative k corresponds to the positive gains, and the positive k corresponds to the negative feedback. None of the script terms are modified, and thus, none of the stability derivatives are changed.

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mp k & 0 \end{bmatrix}$$

Positive Feedback: -k

Figure 11 shows the eigenvalue loci for the requested gain values for *positive feedback*. As k increases, the dutch roll mode has an increasing natural frequency, with slightly better damping. The spiral mode becomes unstable and the roll mode is further stabilized, as explained in many of the previous cases as well.

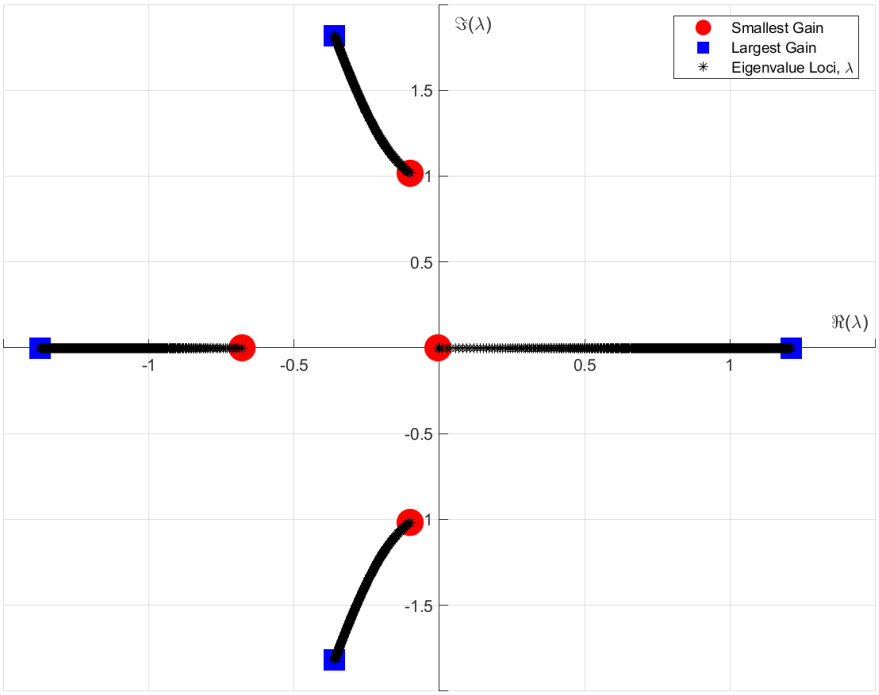


Fig. 11 Part i Positive Feedback Eigenvalue Loci

Negative Feedback: +k

Figure 11 shows the eigenvalue loci for the requested gain values for *negative feedback*. As k increases, the dutch roll mode becomes unstable. This is through the dutch roll mode becoming less damped, with a lower natural frequency. The roll mode and the spiral modes move towards one another, eventually breaking out into an oscillatory mode. This mode then breaks in, and one mode is ultimately further stabilized (more negative Real part), while the other becomes unstable with a positive Real part.

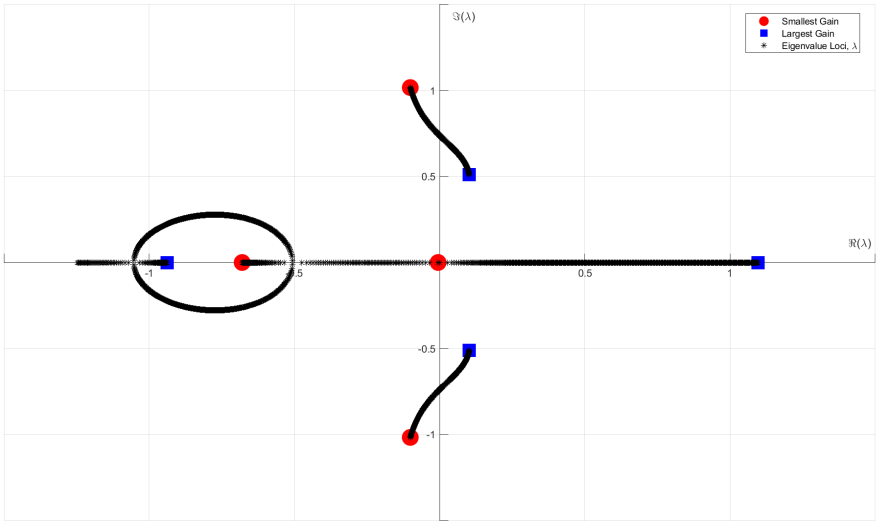


Fig. 12 Part i Negative Feedback Eigenvalue Loci

Overall Analysis

Usually, there is a trade off between Dutch Roll and Spiral stability. As the Dutch Roll is further stabilized (or damped), the Spiral mode tends to become more unstable, or vice versa. In a similar vein, spiral stability is often traded for decreased roll stability. However, this trade-off is usually beneficial. In some cases, these modes even break out as a new oscillatory mode. Thus, it is difficult to improve both of these modes through a system of gains, although the following considerations should be weighed.

Using the yaw rate deviation Δr to control the rudder deflection $\Delta\delta_r$, yields promising results for *positive feedback*. The spiral mode becomes more stable, and the dutch roll mode becomes more damped. Thus, this control should *always* be implemented for increased lateral dynamic stability. Refer to Fig. 7 to see the visualization of this. The roll mode becomes a little less stable, although that is much less of a problem than the spiral or dutch roll modes.

Similarly, positive feedback of the bank angle $\Delta\phi$ should be used to control the aileron deflection $\Delta\delta_r$, as shown in Fig. 1. The spiral mode is once again further stabilized (at the cost of the roll stability). The dutch roll mode is slightly damped as a result. Thus, this control should be usually implemented, but is less significant than the aforementioned method.

As shown by Fig. 6, using negative feedback of the roll rate deviation Δp to control the rudder deflection $\Delta\delta_r$ damps the dutch roll mode *very significantly*, but at the cost of the spiral mode. So very small gains for this control may be used in some scenarios, if the dutch roll mode is more of a problem than the spiral mode. This may be combined with the previous control to further stabilize both the spiral and dutch roll modes.

Negative feedback of the azimuth angle $\Delta\psi$ may be used to control $\Delta\delta_r$, yielding very significant spiral stabilization. However, this destabilizes the dutch roll mode, so it should be implemented after the dutch roll mode has been further stabilized.

In sum, it is a difficult balance to improve lateral stability. The previous eigenvalue loci plots show the results of several control setups, streamlining analysis and allowing for some configurations to be ruled out.

MATLAB

Table of Contents

Assignment 11

```
clc; clear all; close all;
save = 0;
```

From assignment 10:

conversions and constants

```
ft2m = 0.3048;
lb2kg = 0.453592;
slug2kg = 14.5939;
lbf2N = 4.44822;
g = 9.81;

% Non-dimensional Derivatives (Table 6.6, page 187)
Cy = [-0.8771 0 0]';
Cl = [-0.2797 -0.3295 0.304]';
Cn = [0.1946 -0.04073 -0.2737]';

% Flight Conditions: Table E.1, Case II
% Table E.1 Case 2 SI conversions
altitude = 20000*ft2m; % m
M = 0.5;
V = 518*ft2m; % m/s
m = 6.366*10^5*lb2kg; % kg
W = m*g; % N
I_p = [1.82*10^7; 3.31*10^7; 4.97*10^7; 9.70*10^5] * slug2kg *
 ft2m^2; % kg m^2
zeta = deg2rad(-6.8); % rad
CD = 0.04;
[~, ~, ~, rho] = atmosisa(altitude); % kg/m^3
S = 5500 * ft2m^2; % m^2
b = 59.64; % m

u0 = V;
theta0 = deg2rad(0);
```

```
% Dimensionalized Derivatives
% Using Table 4.5:
Yv = 1/2*rho*u0*S*Cy(1);
Yp = 1/4*rho*u0*b*S*Cy(2);
Yr = 1/4*rho*u0*b*S*Cy(3);

Lv = 1/2*rho*u0*b*S*Cl(1);
Lp = 1/4*rho*u0*b^2*S*Cl(2);
Lr = 1/4*rho*u0*b^2*S*Cl(3);

Nv = 1/2*rho*u0*b*S*Cn(1);
Np = 1/4*rho*u0*b^2*S*Cn(2);
Nr = 1/4*rho*u0*b^2*S*Cn(3);

% Convert to stability frame
% B.12,3
I = diag(I_p(1:3));
I(3,1) = I_p(4);
I(1,3) = I_p(4);
L21 = [cos(zeta) 0 -sin(zeta); 0 1 0; sin(zeta) 0 cos(zeta)];
I = L21'*I*(L21);

% B.12,7
Yv1 = Yv;
Yp1 = Yp*cos(zeta) - Yr*sin(zeta);
Yr1 = Yr*cos(zeta) + Yp*sin(zeta);
Lv1 = Lv*cos(zeta) - Nv*sin(zeta);
Lp1 = Lp*cos(zeta)^2 - (Lr+Np)*sin(zeta)*cos(zeta) + Nr*sin(zeta)^2;
Lr1 = Lr*cos(zeta)^2 - (Nr-Lp)*sin(zeta)*cos(zeta) - Np*sin(zeta)^2;
Nv1 = Nv*cos(zeta) + Lv*sin(zeta);
Np1 = Np*cos(zeta)^2 - (Nr-Lp)*sin(zeta)*cos(zeta) - Lr*sin(zeta)^2;
Nr1 = Nr*cos(zeta)^2 + (Lr+Np)*sin(zeta)*cos(zeta) + Lp*sin(zeta)^2;

vNames = {'Y','L', 'N'};
rowNames = ['v', 'p', 'r'];
Tab = table([Yv1; Yp1; Yr1], [Lv1; Lp1; Lr1], [Nv1; Np1;
 Nr1], 'variableNames', vNames);
disp(Tab)

Ixp = (I(1,1)*I(3,3) - I(3,1)^2)/I(3,3);
Izp = (I(1,1)*I(3,3) - I(3,1)^2)/I(1,1);
Izxp = I(3,1)/(I(1,1)*I(3,3) - I(3,1)^2);
%
A_lat = [...
    Yv1/m       Yp1/m    (Yr1/m-u0)        g*cos(theta0);...
    Lv1/Ixp+Izxp*Nv1 Lp1/Ixp+Izxp*Np1 Lr1/Ixp+Izxp*Nr1 0;...
    Izxp*Lv1+Nv1/Izp Izxp*Lp1+Np1/Izp Izxp*Lr1+Nr1/Izp 0;...
    0       1    tan(theta0)     0];
disp(A_lat)

%
[V,D] = eig(A_lat);
V
```

```
lambdas = diag(D);
disp(lambdas);
taus = (-ones(4,1)./real(lambdas));
disp(taus);

wn = norm(lambdas(1));
disp(wn);

zeta = -real(lambdas(1))/wn;
disp(zeta);

%
A_lat_DR = [A_lat(1,1) A_lat(1,3);...
                A_lat(3,1) A_lat(3,3)];

[V_dr, D_dr] = eig(A_lat_DR)
lambdas = diag(D_dr);
wn = norm(lambdas(1));
disp(wn);

zeta = -real(lambdas(1))/wn;
disp(zeta);
```

```
            Y                   L                   N
    ________________    ________________    ________________

    -23092.2461610012    -399915.622189494     355412.472234917
                    0    -13942339.9779819     -1772752.4665956
                    0     14368655.7379063    -14301495.5714496

   1.0e+02 *

  Columns 1 through 3

  -0.000799713003510                   0  -1.578864000000000
  -0.000169548046145  -0.005591825038775   0.006126246567892
   0.000062430339190   0.000047732637240  -0.002473303798611
                   0   0.010000000000000                   0

  Column 4

   0.098100000000000
                   0
                   0
                   0


V =

  Column 1

  0.999759842600857 + 0.000000000000000i
 -0.009025914671038 + 0.012051081183452i
  0.000915750516229 - 0.005964393701972i
```

```
   0.012609053170628 + 0.007626310790283i

   Column 2

   0.999759842600857 + 0.000000000000000i
  -0.009025914671038 - 0.012051081183452i
   0.000915750516229 + 0.005964393701972i
   0.012609053170628 - 0.007626310790283i

   Column 3

  -0.927181328058734 + 0.000000000000000i
  -0.210389743091235 + 0.000000000000000i
   0.015712039082749 + 0.000000000000000i
   0.309554636096939 + 0.000000000000000i

   Column 4

   0.921143947394905 + 0.000000000000000i
  -0.001980952582381 + 0.000000000000000i
   0.023701629428858 + 0.000000000000000i
   0.388494706227788 + 0.000000000000000i

  -0.100866059083649 + 1.016754939861699i
  -0.100866059083649 - 1.016754939861699i
  -0.679653019395744 + 0.000000000000000i
  -0.005099046526568 + 0.000000000000000i

    1.0e+02 *

    0.099141377097988
    0.099141377097988
    0.014713390089682
    1.961150961831133

    1.021745843939790

    0.098719324068612


V_dr =

   Column 1

   0.999980229934998 + 0.000000000000000i
   0.000529987924262 - 0.006265688465633i

   Column 2

   0.999980229934998 + 0.000000000000000i
   0.000529987924262 + 0.006265688465633i


D_dr =
```

```
  Column 1

 -0.163650840106069 + 0.989286553619800i
  0.000000000000000 + 0.000000000000000i

  Column 2

  0.000000000000000 + 0.000000000000000i
 -0.163650840106069 - 0.989286553619800i

   1.002731012106618

   0.163205124934012
```

Question 1: Spiral and Roll Mode Approximations

Spiral

```
sLv1 = A_lat(2,1);
sNr1 = A_lat(3,3);
sLr1 = A_lat(2,3);
sNv1 = A_lat(3,1);
sLp1 = A_lat(2,2);
sNp1 = A_lat(3,2);



e = g * ( (sLv1*sNr1-sLr1*sNv1)*cos(theta0) + (sLp1*sNv1-
sLv1*sNp1)*sin(theta0) );
d = -g*(sLv1*cos(theta0) + sNv1*sin(theta0)) + u0*(sLv1*sNp1-
sLp1*sNv1);

lamSpAppx = -e/d

% Roll
lamRollAppx = sLp1


lamSpAppx =

  -0.005133806132668


lamRollAppx =

  -0.559182503877474
```

Question 2:

Using the flight conditions from problem 1 of assignment 10 and the non-dimensionalcontrol coefficients in Table 7.3, find the lateral control derivatives for B747 aircraft. Use these to construct the 4x2 lateralB-matrix (i.e. the control sensitivity matrix) forthe linearized dynamics model.

```
% Table 7.3 Page 243
Cy_d = [0; 0.1146];
Cl_d = [-1.368e-2; 6.976e-3];
Cn_d = [-1.973e-4; -0.1257];

% Dimensionalized Derivatives
% Using Table 4.1 dimensionalization factors
Yda = 1/2*rho*u0^2*S*Cy_d(1);
Ydr = 1/2*rho*u0^2*S*Cy_d(2);

Lda = 1/2*rho*u0^2*b*S*Cl_d(1);
Ldr = 1/2*rho*u0^2*b*S*Cl_d(2);

Nda = 1/2*rho*u0^2*b*S*Cn_d(1);
Ndr = 1/2*rho*u0^2*b*S*Cn_d(2);

B_ctr = [Yda/m Ydr/m;...
    Lda/Ixp+Izxp*Nda Ldr/Ixp+Izxp*Ndr;...
    Izxp*Lda+Nda/Izp Izxp*Ldr+Ndr/Izp;...
    0 0];

disp('Question 2');
disp(B_ctr);

% Find A lat aug matrix and B aug
A_aug = zeros(6);
A_aug(1:4,1:4) = A_lat;
A_aug(6,1) = 1;
A_aug(5,3) = sec(theta0);
A_aug(6,5) = u0*cos(theta0);

B_aug = zeros(6,2);
B_aug(1:4,1:2) = B_ctr;
% delta a
% delta r

disp(A_aug);
disp(B_aug);
```

```
Question 2
                   0   1.649735754216573
  -0.136881697062276   0.140080102843713
   0.006907156779869  -0.472273091566520
                   0                   0

   1.0e+02 *
```

```
  Columns 1 through 3

  -0.000799713003510                   0  -1.578864000000000
  -0.000169548046145  -0.005591825038775   0.006126246567892
   0.000062430339190   0.000047732637240  -0.002473303798611
                   0   0.010000000000000                   0
                   0                   0   0.010000000000000
   0.010000000000000                   0                   0

  Columns 4 through 6

   0.098100000000000                   0                   0
                   0                   0                   0
                   0                   0                   0
                   0                   0                   0
                   0                   0                   0
                   0   1.578864000000000                   0

                   0   1.649735754216573
  -0.136881697062276   0.140080102843713
   0.006907156779869  -0.472273091566520
                   0                   0
                   0                   0
                   0                   0
```

Question 3

y_aug = [v, p, r, phi, psi, yE] Positive values for negative feedback, Negative values for positive feedback

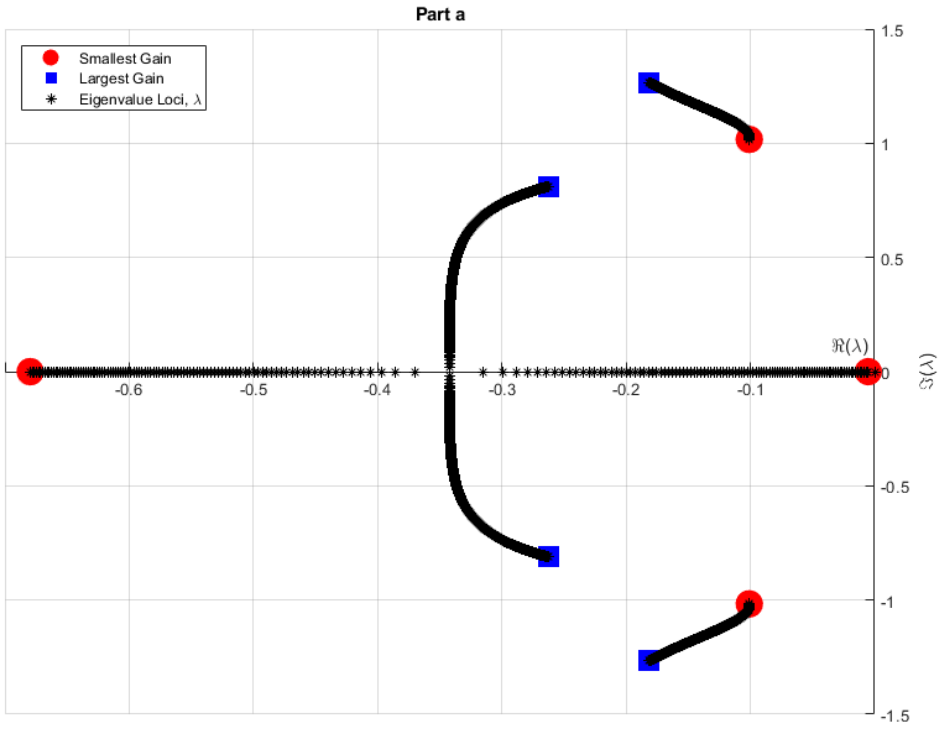
a-d

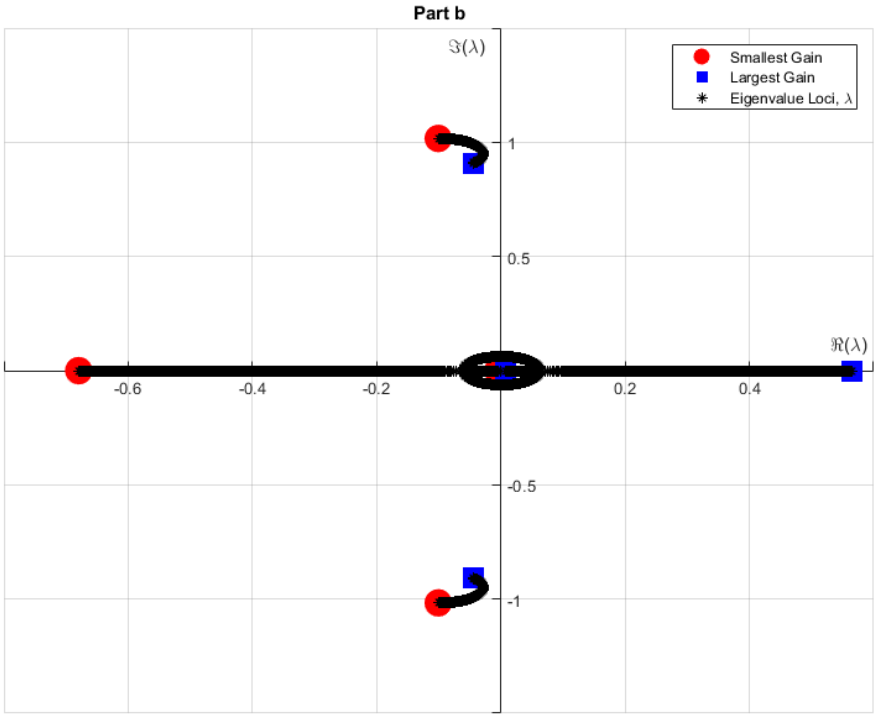
a

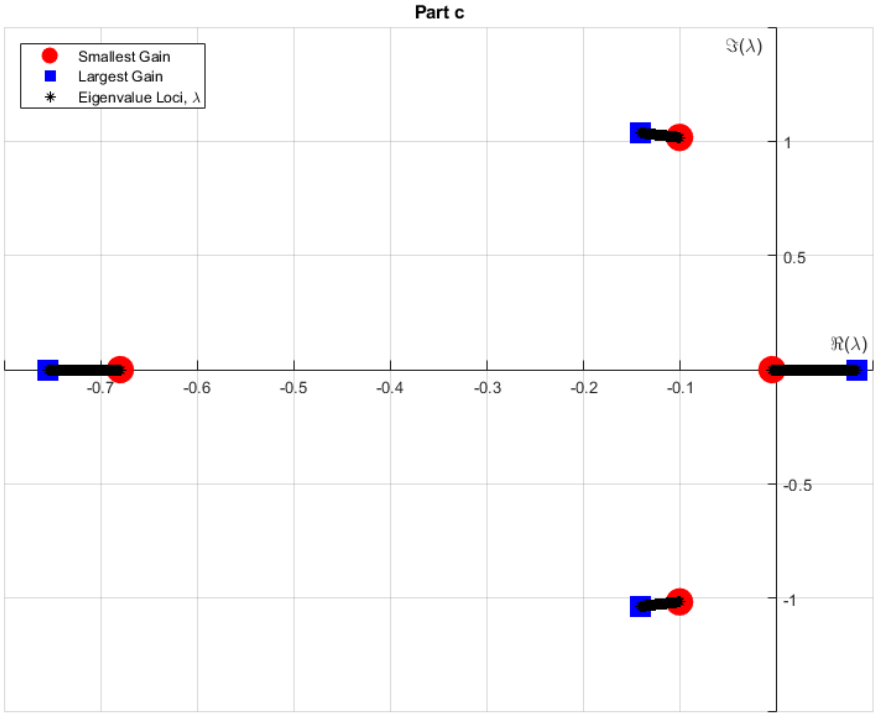
```
k = 0:0.01:10;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 0 0 -k(i) 0 0; 0 0 0 0 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'a.png');
end
title('Part a');

% b
```

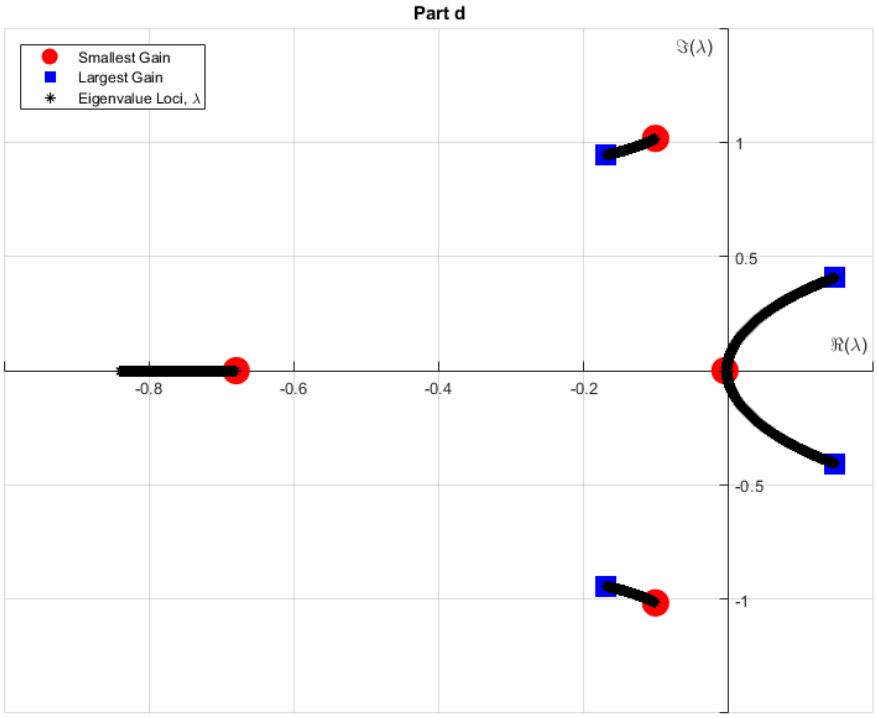
```
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:10;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 k(i) 0 0 0 0; 0 0 0 0 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'b.png');
end
title('Part b');

% c
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:10;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 0 k(i) 0 0 0; 0 0 0 0 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'c.png');
end
title('Part c');

% d
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:20;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 0 0 0 -k(i) 0; 0 0 0 0 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'d.png');
end
```

```
title('Part d');
```


Part a
Smallest Gain
Largest Gain
Eigenvalue Loci, λ
1.5
1
0.5
0
-0.5
-1
-1.5
-0.6
-0.5
-0.4
-0.3
-0.2
-0.1
ℜ(λ)
ℑ(λ)


Part b
Smallest Gain
Largest Gain
Eigenvalue Loci, λ
ℑ(λ)
ℜ(λ)
1
0.5
-0.5
-1
-0.6
-0.4
-0.2
0.2
0.4


Part c
Smallest Gain
Largest Gain
Eigenvalue Loci, λ
ℑ(λ)
ℜ(λ)
1
0.5
-0.5
-1
-0.7
-0.6
-0.5
-0.4
-0.3
-0.2
-0.1



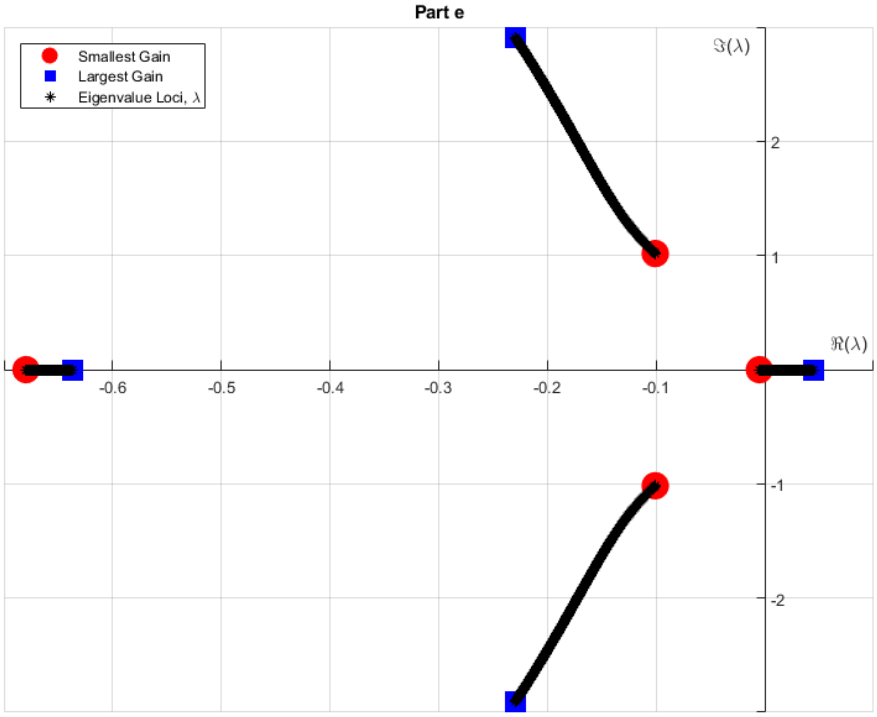
e-g

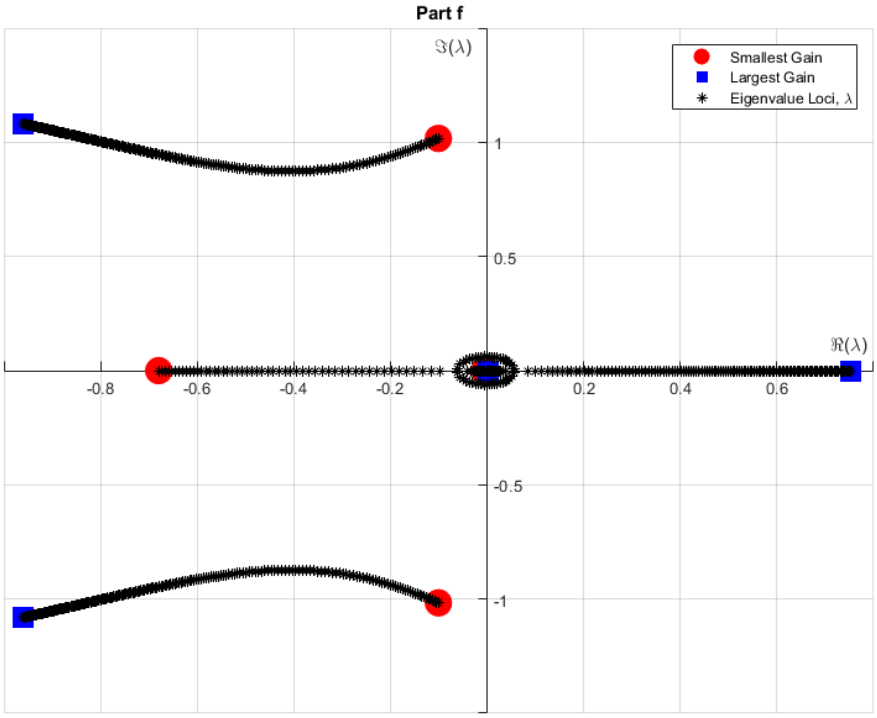
e y_aug = [v, p, r, phi, psi, yE]

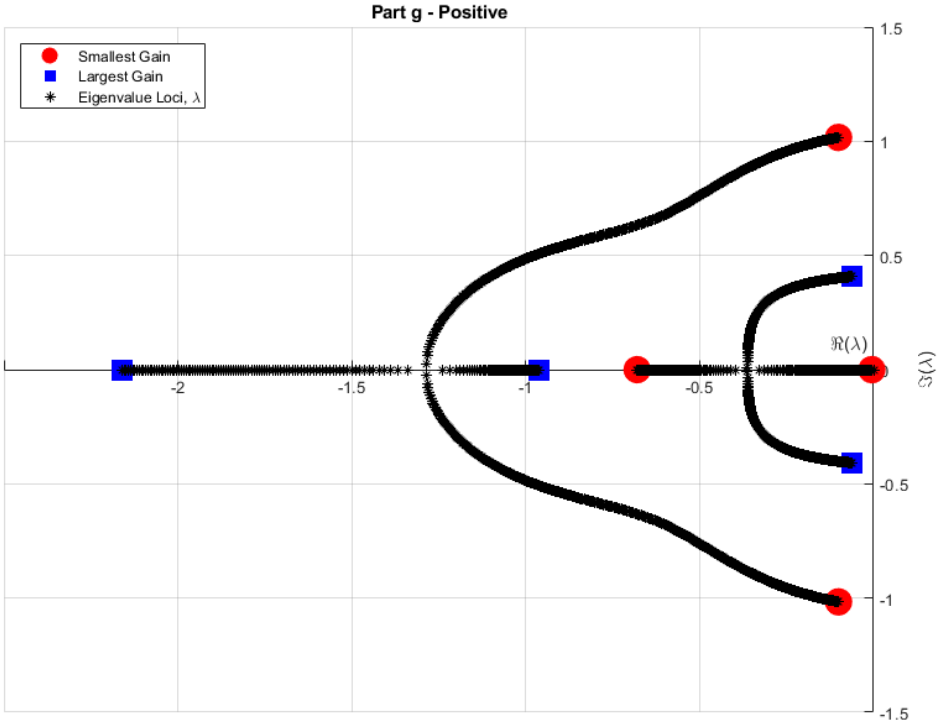
```
k = 0:0.0001:0.1;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 0 0 0 0 0; k(i) 0 0 0 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'e.png');
end
title('Part e');

% f
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:2;
lams = zeros(length(k),6);
```

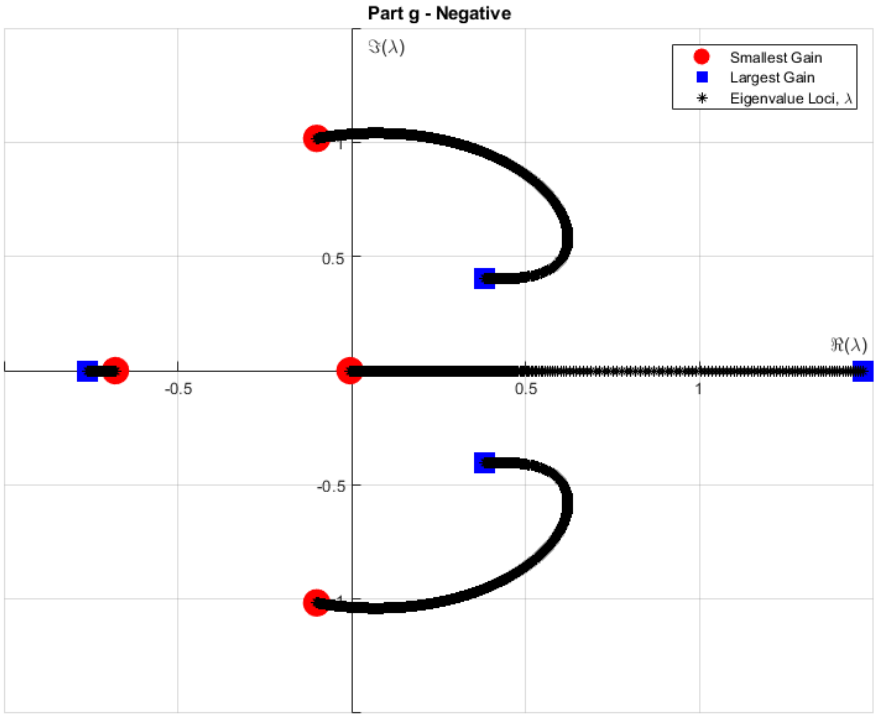
```
for i=1:length(k)
    K_mat = [0 0 0 0 0 0; 0 k(i) 0 0 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'f.png');
end
title('Part f');

% g - Pos
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:5;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 0 0 0 0 0; 0 0 -k(i) 0 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'gp.png');
end
title('Part g - Positive');

% g - Neg
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:5;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 0 0 0 0 0; 0 0 k(i) 0 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'gn.png');
end
title('Part g - Negative');

Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.
```

```
Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.
```




Part f
Smallest Gain
Largest Gain
Eigenvalue Loci, λ
ℑ(λ)
ℜ(λ)
1
0.5
-0.5
-1
-0.8
-0.6
-0.4
-0.2
0.2
0.4
0.6


Part g - Positive
Smallest Gain
Largest Gain
Eigenvalue Loci, λ
ℑ(λ)
ℜ(λ)
1.5
1
0.5
-0.5
-1
-1.5
-2
-1.5
-1
-0.5



h-i

```
% h - Neg
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:5;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 0 0 0 0 0; 0 0 0 k(i) 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'hn.png');
end
title('Part h - Negative');

% h - Pos
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:5;
lams = zeros(length(k),6);
```

```
for i=1:length(k)
    K_mat = [0 0 0 0 0 0; 0 0 0 -k(i) 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'hp.png');
end
title('Part h - Positive');

% i - Pos
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:5;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 0 0 0 0 0; 0 0 0 k(i) 0 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
if (save==1)
    saveas(gcf,'ip.png');
end
title('Part i - Positive');

% i - Neg
% y_aug = [v, p, r, phi, psi, yE]
k = 0:0.01:5;
lams = zeros(length(k),6);

for i=1:length(k)
    K_mat = [0 0 0 0 0 0; 0 0 0 0 k(i) 0];
    A_lat_CL = A_aug - B_aug*K_mat;
    [~,D] = eig(A_lat_CL);
    lams(i,:) = diag(D);
end

f = figure();
plotEVs(lams,f);
plotEVs(conj(lams),f);
if (save==1)
    saveas(gcf,'in.png');
end
title('Part i - Negative');
```

Functions

```
function f = plotEVs(lams,f)
    hold on;

    h1 =
 plot(real(lams(1,3:6)),imag(lams(1,3:6)),'or','MarkerSize',16,'MarkerFaceColor','
    h2 =
 plot(real(lams(end,3:6)),imag(lams(end,3:6)),'sb','MarkerSize',16,'MarkerFaceColo
    h3 = plot(lams(:,1:6),'*k');
    legend([h1; h2; h3],'Smallest Gain', 'Largest Gain', 'Eigenvalue
 Loci, \lambda','location','best');
    grid on;
    pos = get(gcf,'position');
    set(gcf,'position',[pos(1:2)/4 pos(3:4)*1.62]);
    ax = gca;
    ax.XAxisLocation = 'origin';
    ax.YAxisLocation = 'origin';
    xlabel('\Re(\lambda)');
    ylabel('\Im(\lambda)')
end

function der_y_aug = stateDerAug(y_aug,A_lat_CL)
    % Computes the time rate of change of the augmented y state
 variable
```

```
    % Note that y is assumed to be a zero vector for trim, thus we do
 not need to
    % explicitely subtract the trim vector to find the deviations from
 trim.
    der_y_aug = A_lat_CL*y_aug;
end

function f = plotState(y,t,f)
    names = {'Side-slip Velocity v, m/s','Roll Rate p, rad/s','Yaw
 Rate r, rad/s','Bank Angle \phi, deg'};
    figure(f);
    hold on;

    for i=1:4
        subplot(2,2,i);
        plot(t,y(:,i),'Linewidth',2);
        grid on;
        xlabel('Time, s')
        ylabel(names{i});
    end

end
```

```
Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.
```

Published with MATLAB® R2019b