AAE 364: Control Systems Analysis

HW12: Bode, Nyquist Plot & Practice

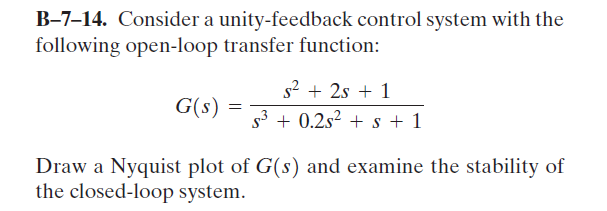
Dr. Sun

School of Aeronautical and Astronautical

Purdue University

Tomoki Koike

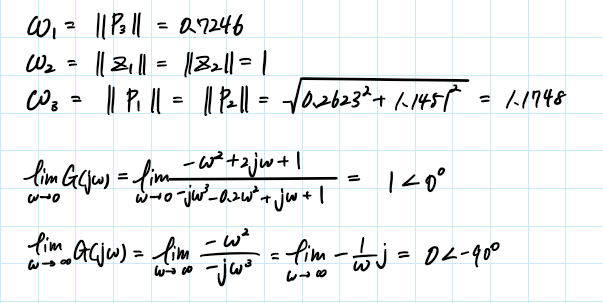
Friday May 1st, 2020



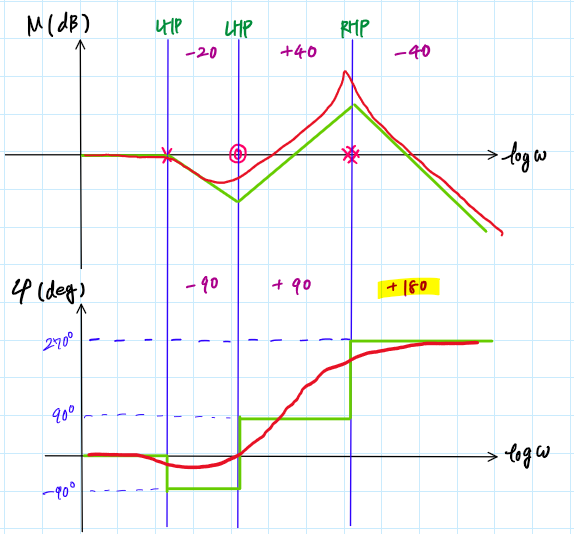
Poles & Zeros

|  |  |  |
| --- | --- | --- |
|  | Poles, | Zeros, |
| 1 | 0.2623+1.1451j | -1 |
| 2 | 0.2623-1.1451j | -1 |
| 3 | -0.7246 |  |

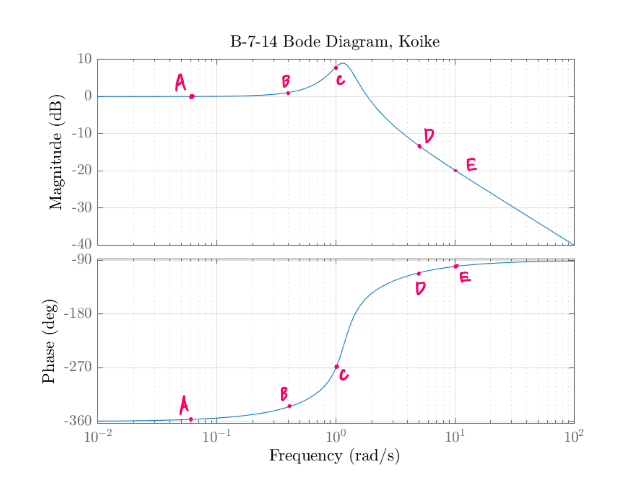
Corner Frequencies & Limits



Bode Plot Sketch



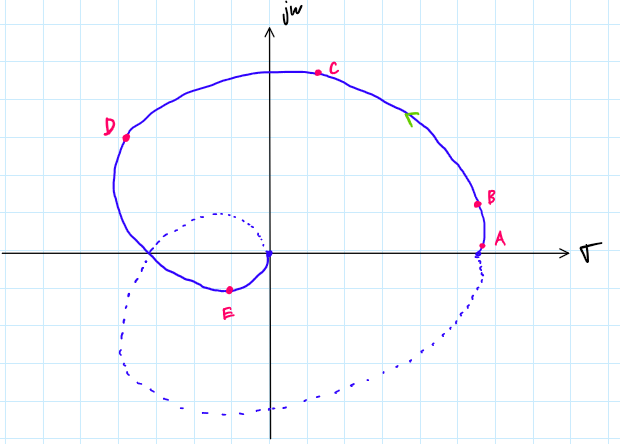
Bode Plot (MATLAB)



Sample Points from Bode Plot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Point | [rad/s] | [deg] | [dB] | |G| |
| A | 0.05 | -357.1318766 | -0.015231146 | 1.001755089 |
| B | 0.4 | -335.5394486 | -1.07759798 | 1.132087249 |
| C | 1 | -269.9999453 | -7.958797179 | 2.499999138 |
| D | 4 | -115.9725883 | 10.95987844 | 0.283143162 |
| E | 10 | -100.3217069 | 19.82787424 | 0.102001437 |

Nyquist Plot Sketch

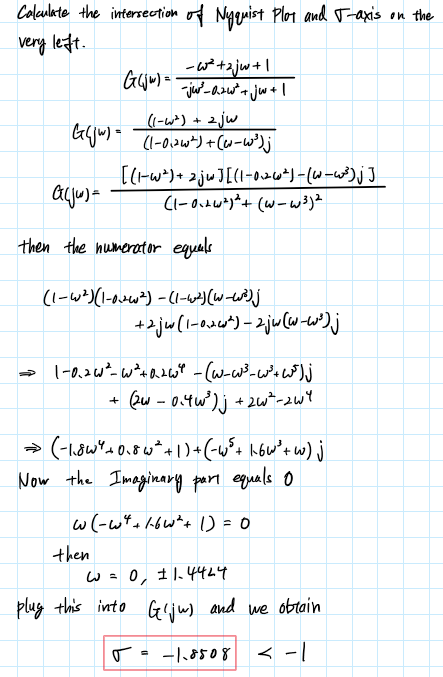


Nyquist Plot (MATLAB)

A close up of a map

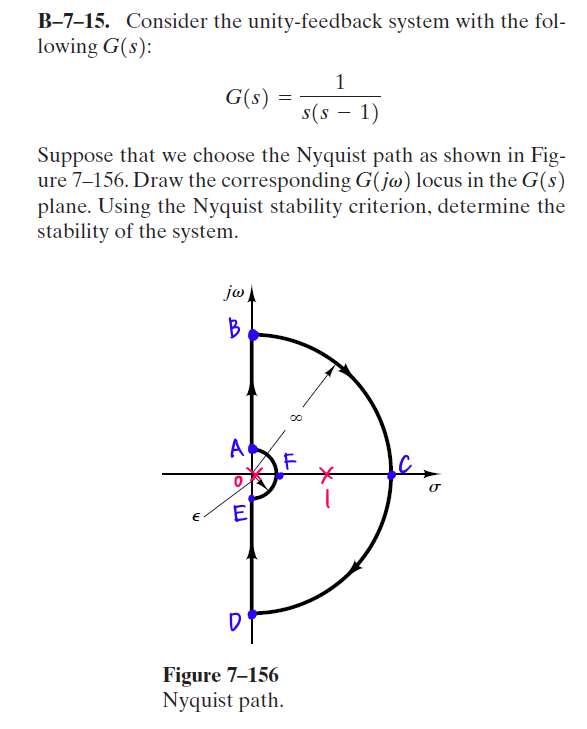
Description automatically generated

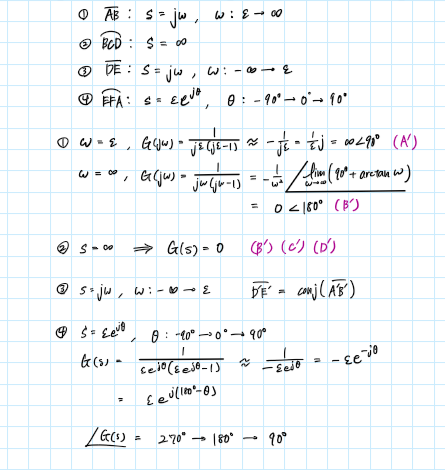
Nyquist Stability Analysis



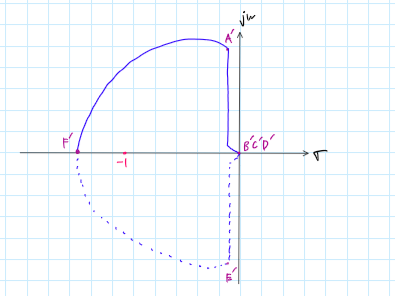
|  |  |  |  |
| --- | --- | --- | --- |
| P: the number of OL poles in RHP | **P** | **N** | **Z** |
| N: the number of clockwise encirclements about -1 | 2 | 2 | 4 |
| Z: the number of CL poles in RHP ⟺ Z = P + N |  | | |

The system is unstable





Nyquist Plot Sketch

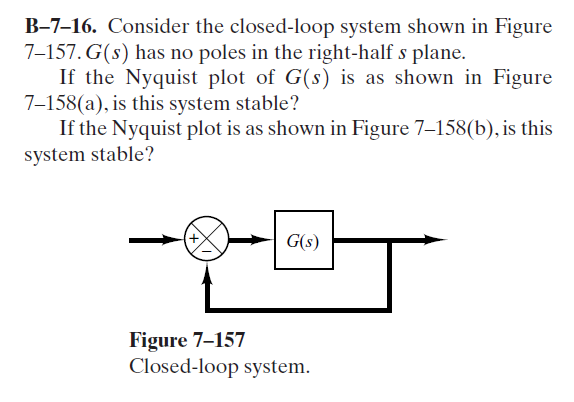


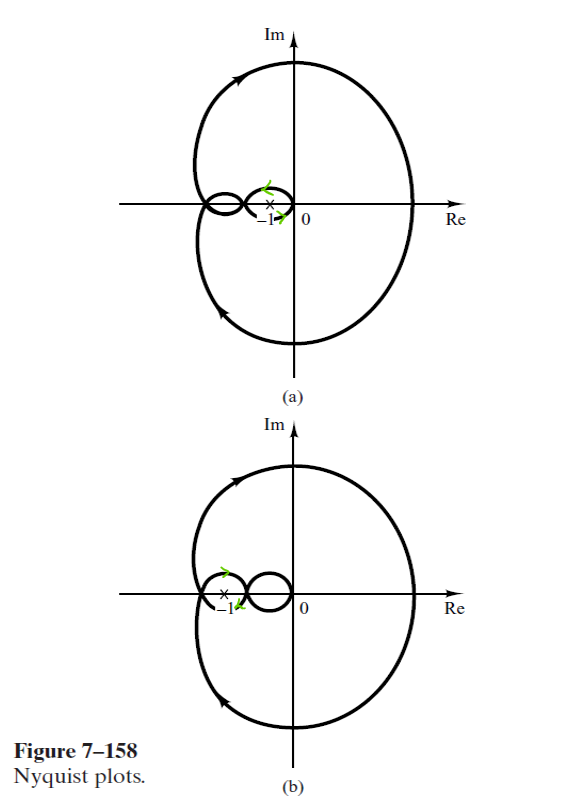


Nyquist Stability Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| P: the number of OL poles in RHP | **P** | **N** | **Z** |
| N: the number of clockwise encirclements about -1 | 1 | 1 | 2 |
| Z: the number of CL poles in RHP ⟺ Z = P + N |  | | |

The system is unstable





7-158(a)

There is one clockwise encirclement outside and one counter-clockwise encirclement inside. Thus, the net clockwise encirclement becomes 0.

|  |  |  |  |
| --- | --- | --- | --- |
| P: the number of OL poles in RHP | **P** | **N** | **Z** |
| N: the number of clockwise encirclements about -1 | 0 | 0 | 0 |
| Z: the number of CL poles in RHP ⟺ Z = P + N |  | | |

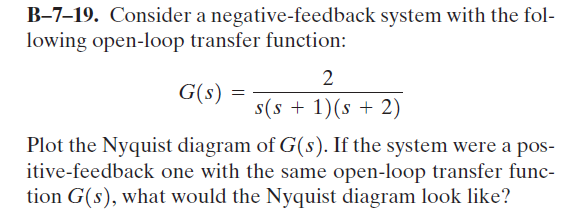
The system is stable

7-158(b)

There is one clockwise encirclement outside and another clockwise encirclement inside. Thus, the net clockwise encirclement becomes 2.

|  |  |  |  |
| --- | --- | --- | --- |
| P: the number of OL poles in RHP | **P** | **N** | **Z** |
| N: the number of clockwise encirclements about -1 | 0 | 2 | 2 |
| Z: the number of CL poles in RHP ⟺ Z = P + N |  | | |

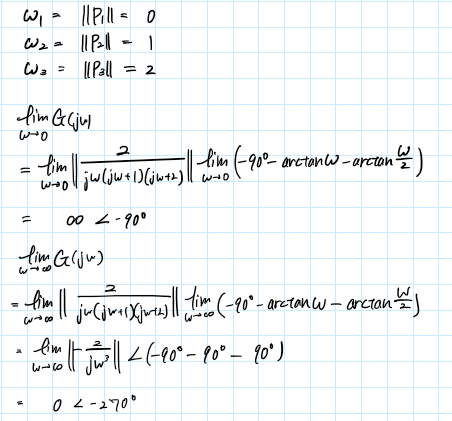
The system is unstable



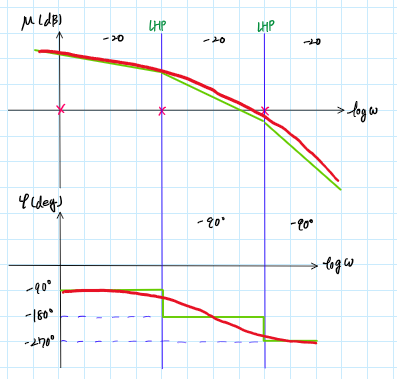
Poles & Zeros

|  |  |  |
| --- | --- | --- |
|  | Poles, | Zeros, |
| 1 | 0 |  |
| 2 | -1 |  |
| 3 | -2 |  |

Corner Frequencies & Limits



Bode Plot Sketch



Bode Plot (MATLAB)

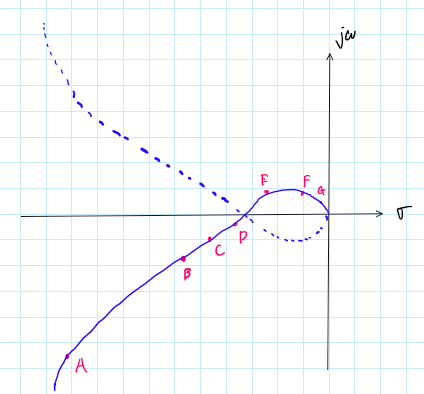
A close up of a map

Description automatically generated

Sample Points from Bode Plot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Point | [rad/s] | [deg] | [dB] | |G| |
| A | 0.05 | -94.29450141 | -26.00704399 | 19.96881068 |
| B | 0.2 | -107.0205255 | -13.76585437 | 4.878571998 |
| C | 0.7 | -144.2820646 | -0.864315168 | 1.104627265 |
| D | 1 | -161.5650472 | 3.979396614 | 0.632455785 |
| E | 4 | -229.3986988 | 31.33538003 | 0.027116335 |
| F | 10 | -252.9794718 | 54.19293923 | 0.00195143 |
| G | 15 | -258.5912795 | 64.6406538 | 0.000586094 |

Nyquist Plot Sketch



Nyquist Plot (MATLAB)

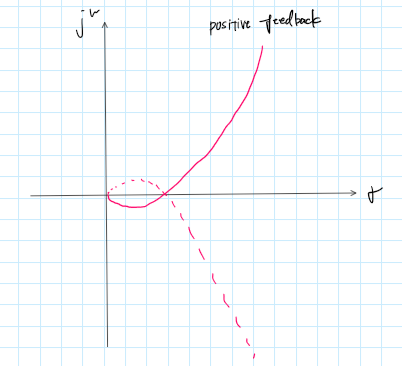
A close up of a map

Description automatically generated

Discussion

If the system were to be a positive feedback system it would change the Nyquist plot by rotating the plot 180o. This is because the Nyquist stability criterion is changed to assess the encirclements to be based on the point [1, 0] and not [-1, 0] because the characteristic equation is changed to the following

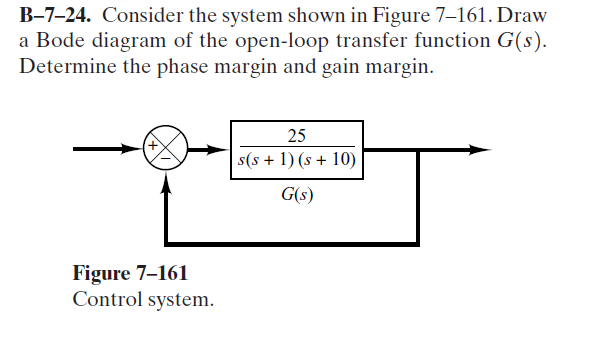
Nyquist Plot Sketch (Positive Feedback)



Nyquist Plot (MATLAB) (Positive Feedback)

A close up of a map

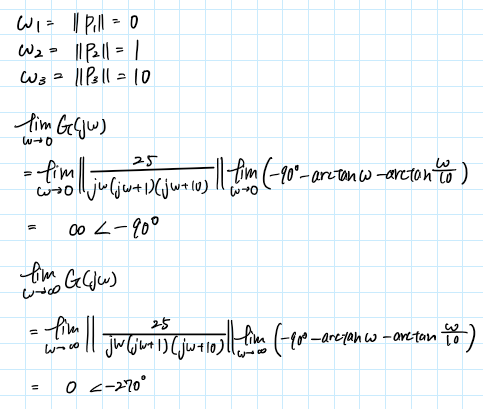
Description automatically generated



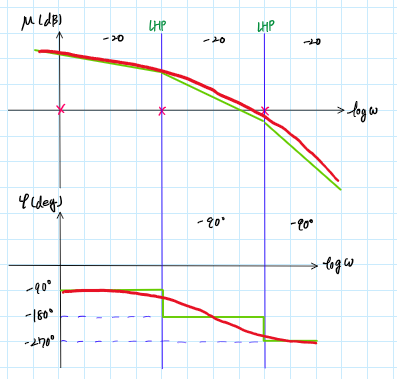
Poles & Zeros

|  |  |  |
| --- | --- | --- |
|  | Poles, | Zeros, |
| 1 | 0 |  |
| 2 | -1 |  |
| 3 | -10 |  |

Corner Frequencies & Limits



Bode Plot Sketch

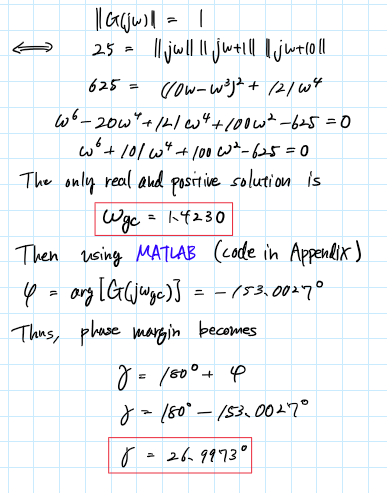


Bode Plot (MATLAB)

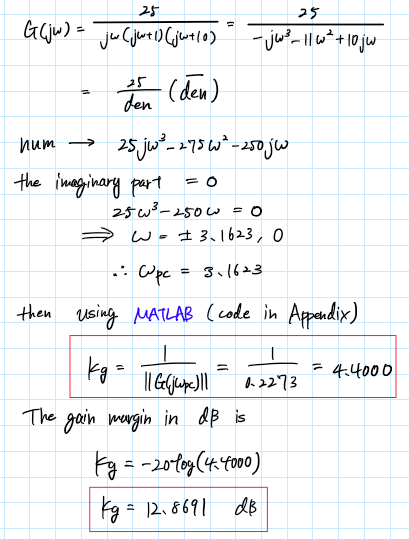
A close up of a map

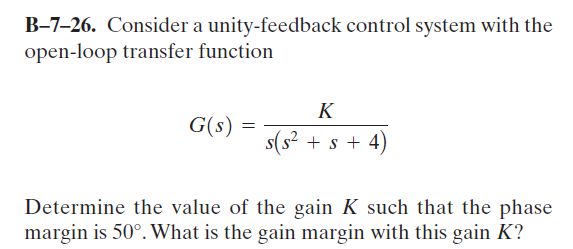
Description automatically generated

Phase Margin

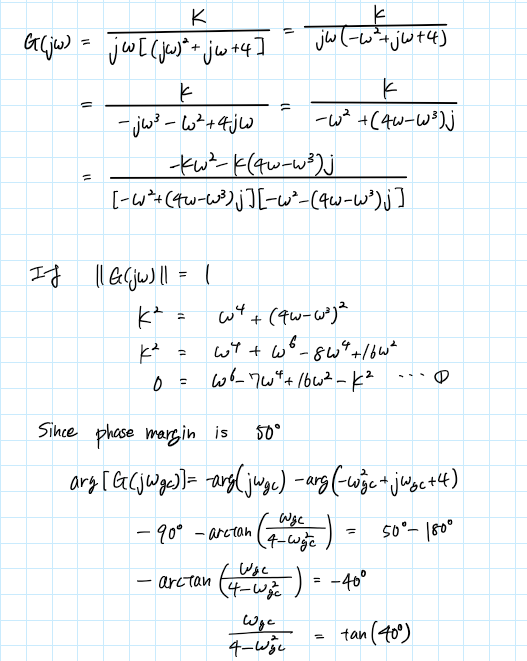


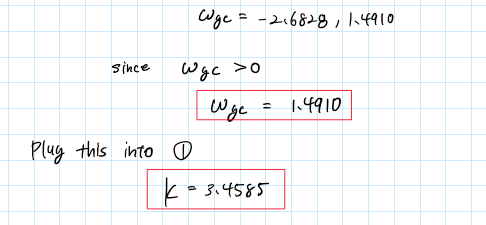
Gain Margin



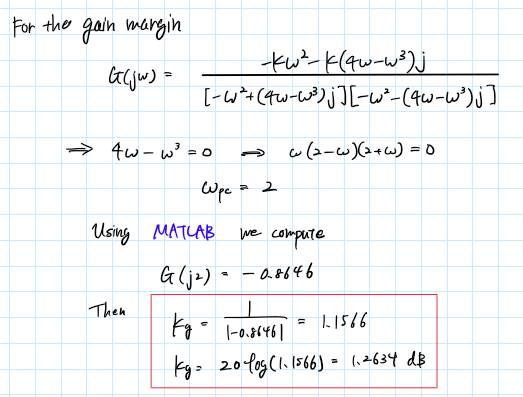


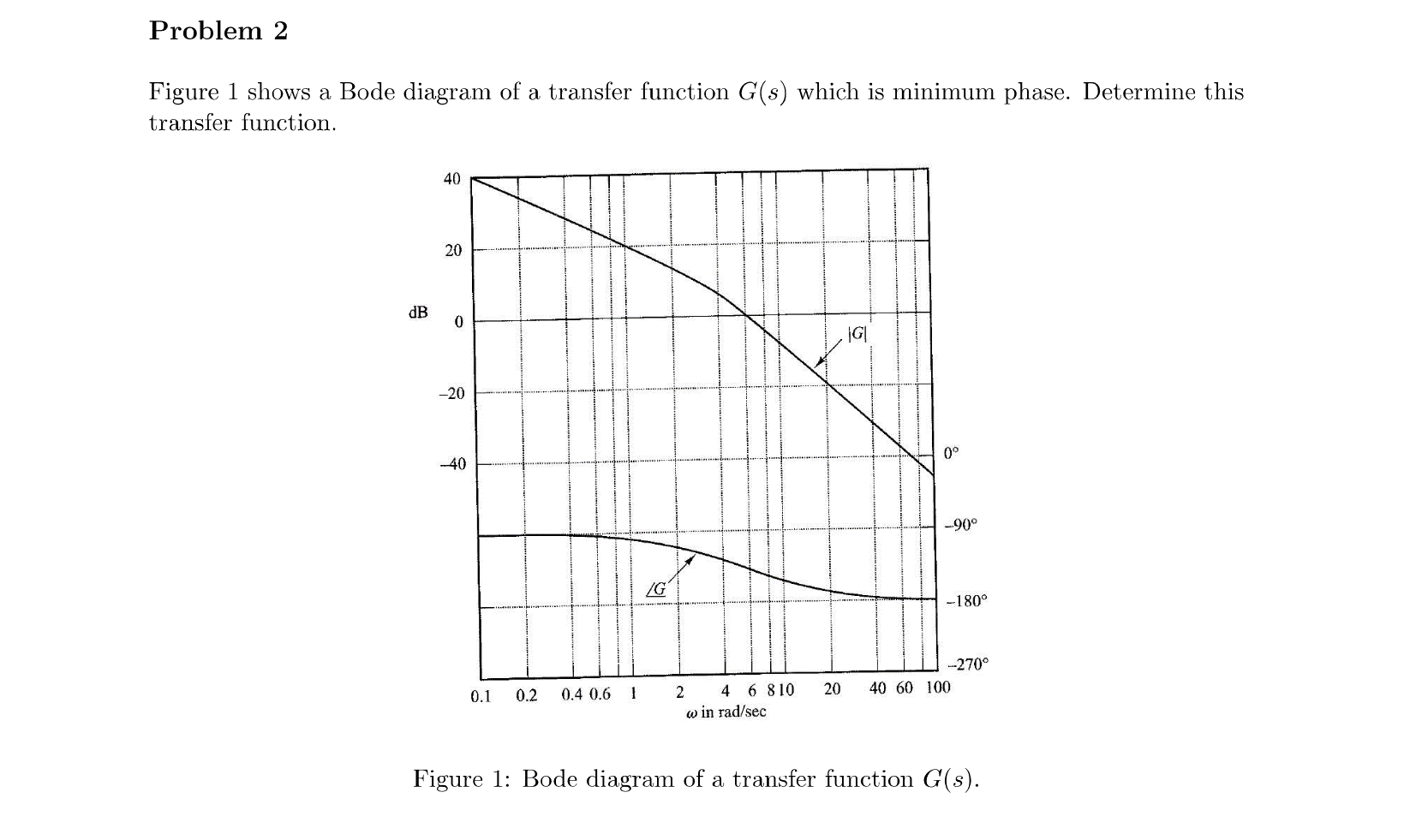
Phase Margin

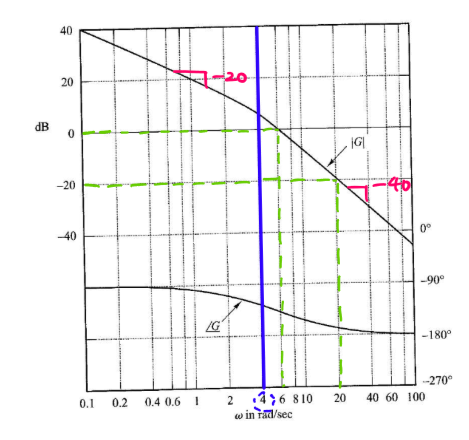


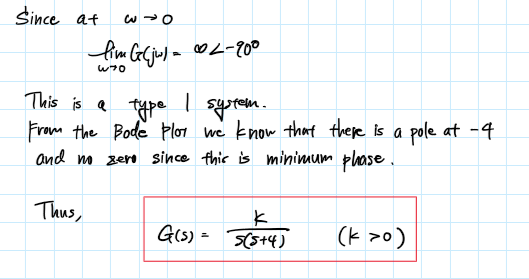


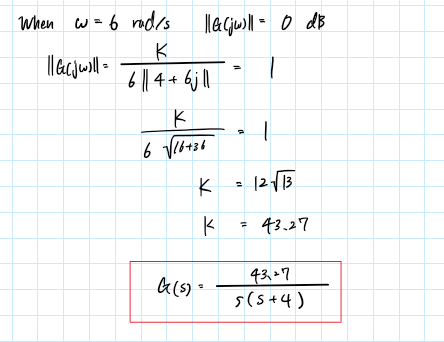
Gain Margin

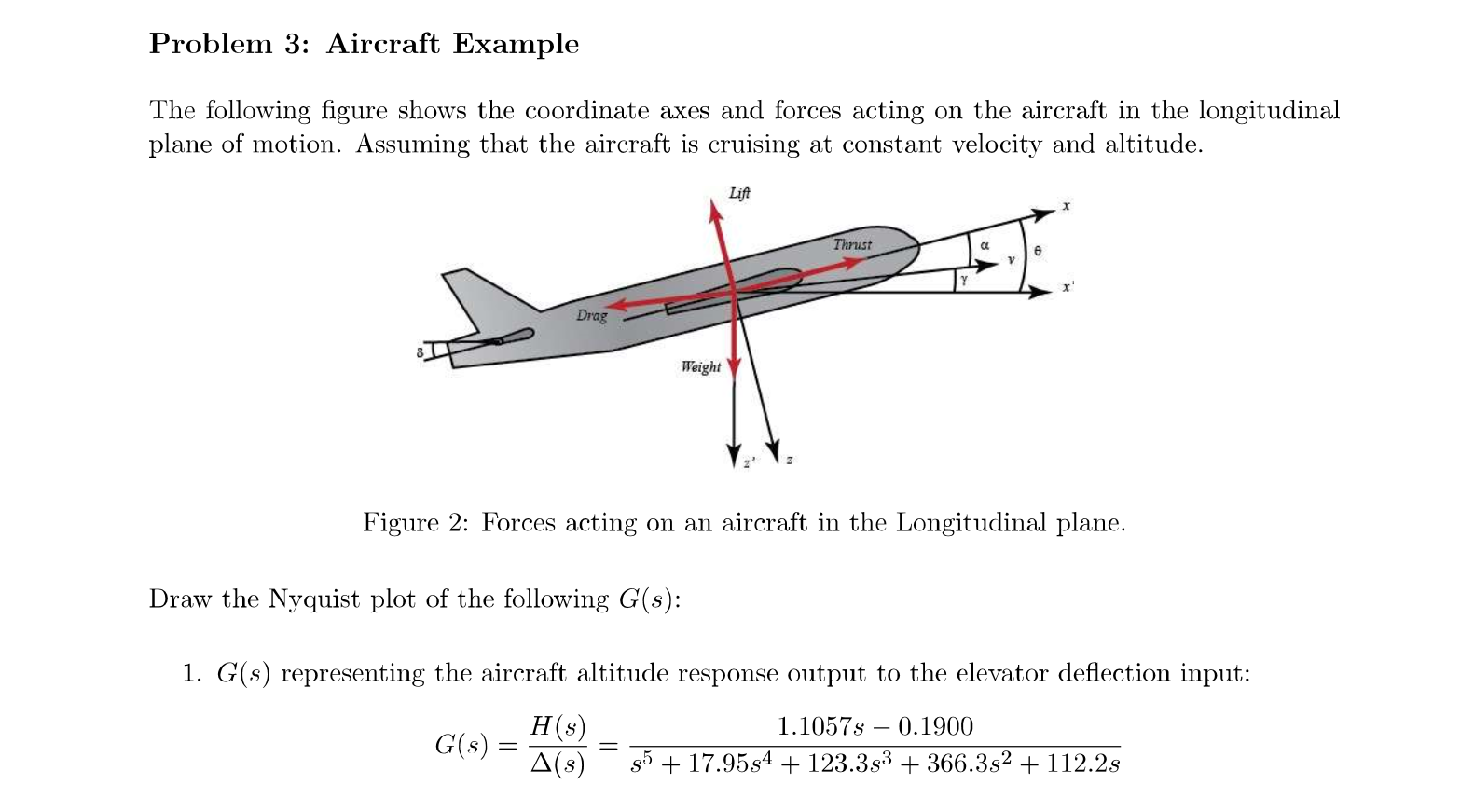




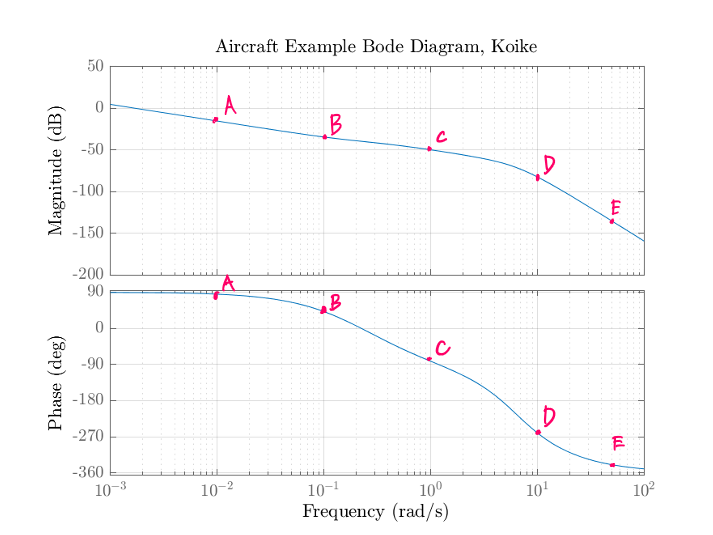








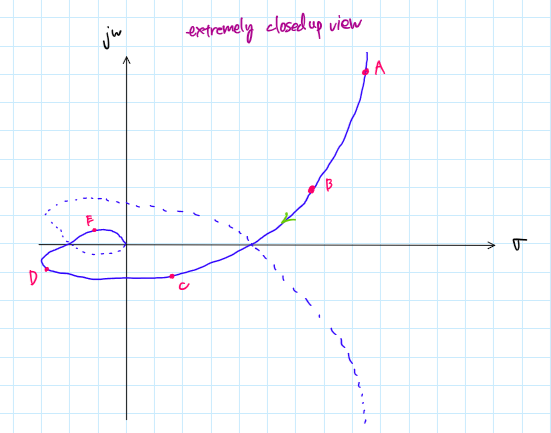
Bode Plot (From HW11)



Sample Points from Bode Plot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Point | [rad/s] | [deg] | [dB] | |G| |
| A | 0.01 | 84.79937283 | 15.41377481 | 0.169555257 |
| B | 0.1 | 41.54326143 | 34.51085838 | 0.018812958 |
| C | 1 | -81.91045187 | 49.8448367 | 0.003219276 |
| D | 10 | -260.191817 | 82.32819474 | 7.64875E-05 |
| E | 50 | -339.2704051 | 135.1757373 | 1.74266E-07 |

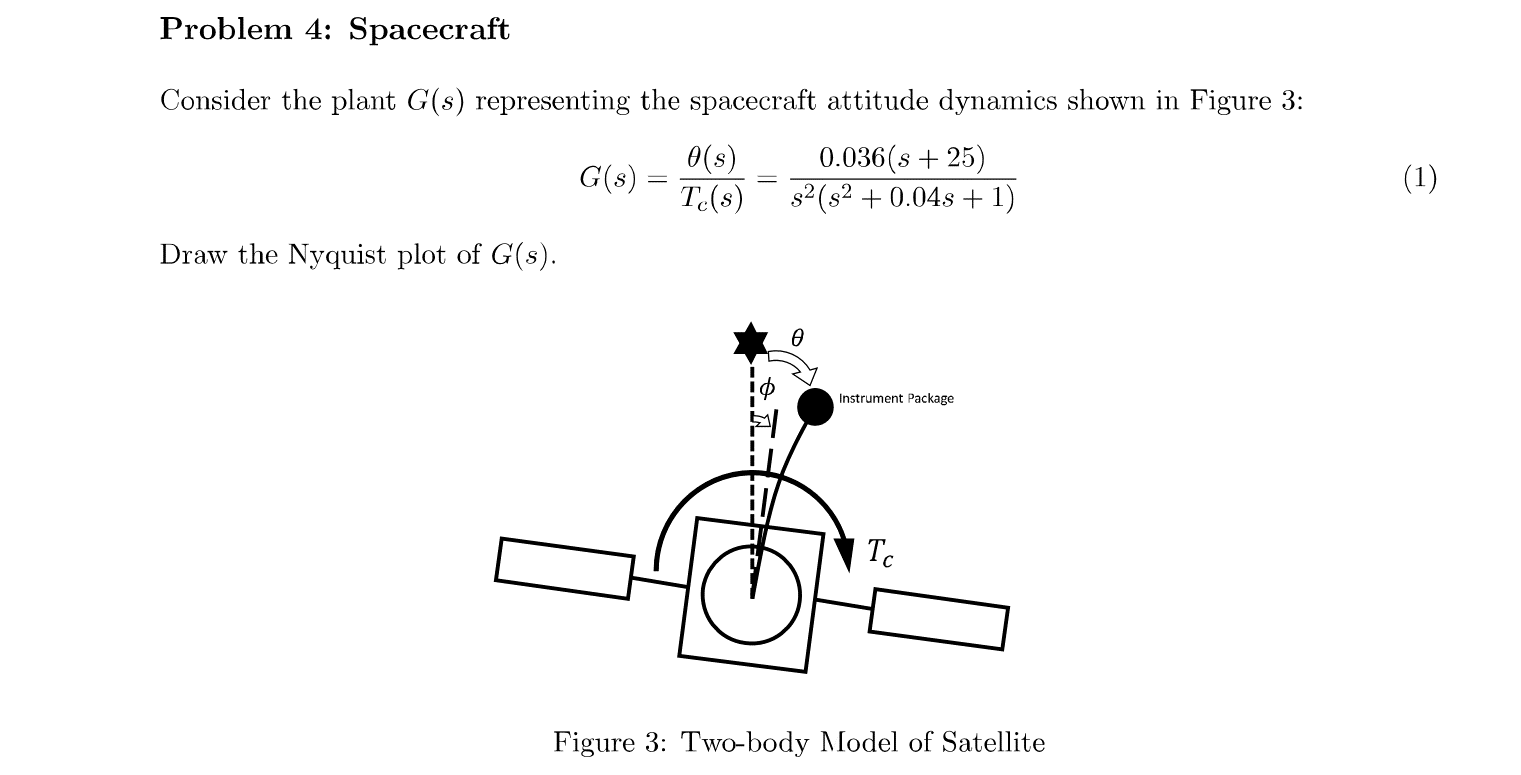
Nyquist Plot Sketch



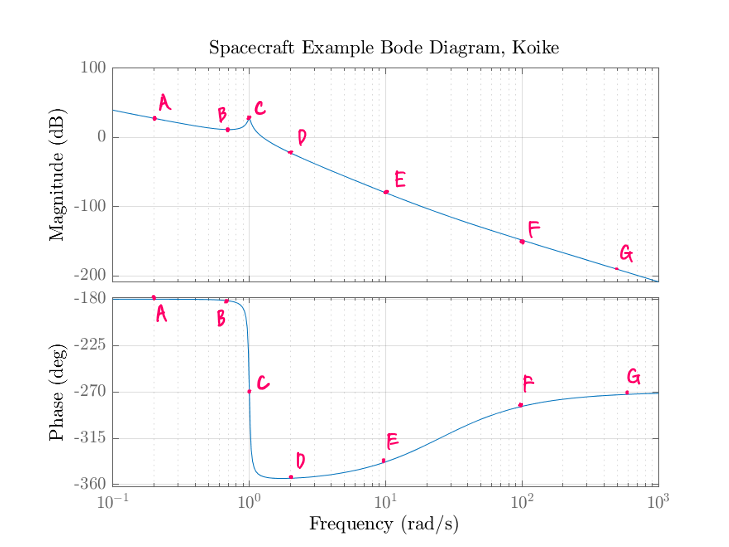
Nyquist Plot (MATLAB)

A close up of a map

Description automatically generated



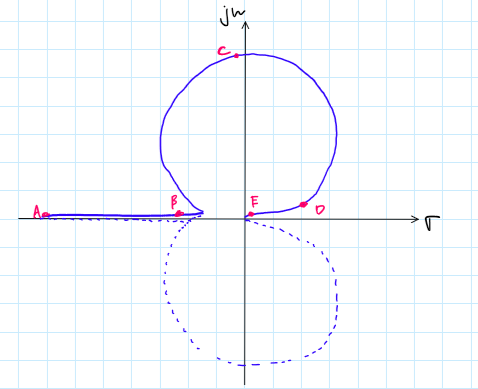
Bode Plot (From HW11)



Sample Points from Bode Plot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Point | [rad/s] | [deg] | [dB] | |G| |
| A | 0.2 | -180.0190973 | -27.39820206 | 23.4374362 |
| B | 0.7 | -181.538633 | -11.11985768 | 3.597434405 |
| C | 1 | -267.7093899 | -27.05059318 | 22.51799191 |
| D | 2 | -353.8985533 | 22.47415567 | 0.07521288 |
| E | 10 | -337.9670937 | 80.18334471 | 9.79113E-05 |
| F | 100 | -284.0133229 | 148.6097927 | 3.71117E-08 |
| G | 600 | -272.3821243 | 195.5554676 | 1.66812E-10 |

Nyquist Plot Sketch



Nyquist Plot (MATLAB)

A close up of a map

Description automatically generated

Appendix

MATLAB CODE

## **AAE 364 HW11**

clear all; close all; clc;

fdir = 'C:\Users\Tomo\Desktop\studies\2020-Spring\AAE364\matlab\matlab\_output\hw12';

set(groot, 'defaulttextinterpreter','latex');

set(groot, 'defaultAxesTickLabelInterpreter','latex');

set(groot, 'defaultLegendInterpreter','latex');

% Bode plot options

opts\_bd = bodeoptions('cstprefs');

opts\_bd.Title.Interpreter = "latex";

opts\_bd.XLabel.Interpreter = "Latex";

opts\_bd.YLabel.Interpreter = "Latex";

opts\_bd.Grid = 'on';

% Nyquist plot options

opts\_nq = nyquistoptions("cstprefs");

opts\_nq.Title.Interpreter = 'latex';

opts\_nq.XLabel.Interpreter = "Latex";

opts\_nq.YLabel.Interpreter = "Latex";

opts\_nq.Grid = 'on';

### B-7-14

% Define transfer function

num = [1 2 1];

den = [1 0.2 1 1];

G = tf(num,den);

% Find zeros, poles, and corner frequencies

zrs = roots(num);

pls = roots(den);

cornFreq = corner\_freq(num,den);

% Bode Plot

fig = figure("Renderer","painters");

opts\_bd.Title.String = "B-7-14 Bode Diagram, Koike";

bd = bodeplot(G,opts\_bd);

opt = getoptions(bd);

saveas(fig,fullfile(fdir,"B-7-14\_bode.png"));

% Sample out points from Bode plot

res = bode\_sample\_points(G,[0.05,0.4,1,4,10]);

writetable(res,fullfile(fdir,"B-7-14.xls"),"WriteMode","overwrite");

% Nyquist Plot

fig = figure("Renderer","painters");

opts\_nq.Title.String = "B-7-14 Nyquist Diagram, Koike";

nyquistplot(G,opts\_nq);

saveas(fig,fullfile(fdir,"B-7-14\_nyquist.png"));

% Calculate the intersection with the real axis

[Wpc,intrsct\_Re] = phase\_crossover(num,den);

### B-7-19

% Define transfer function

num = [0 2];

den = conv([1 0],[1 1]); den = conv(den,[1 2]);

G = tf(num,den);

% Find zeros, poles, and corner frequencies

zrs = roots(num);

pls = roots(den);

cornFreq = corner\_freq(num,den);

% Bode Plot

fig = figure("Renderer","painters");

opts\_bd.Title.String = "B-7-19 Bode Diagram, Koike";

bd = bodeplot(G,opts\_bd);

opt = getoptions(bd);

saveas(fig,fullfile(fdir,"B-7-19\_bode.png"));

% Sample out points from Bode plot

res = bode\_sample\_points(G,[0.05,0.2,0.7,1,4,10,15]);

writetable(res,fullfile(fdir,"B-7-19.xls"),"WriteMode","overwritesheet");

% Nyquist Plot

fig = figure("Renderer","painters");

opts\_nq.Title.String = "B-7-19 Nyquist Diagram, Koike";

nyquistplot(G,opts\_nq);

saveas(fig,fullfile(fdir,"B-7-19\_nyquist.png"));

% Nyquist Plot 2

G1 = zpk([],[0, -1, -2],2); % Negative Feedback

G2 = zpk([],[0, -1, -2],-2); % Positive Feedback

fig = figure("Renderer","painters");

hold on; grid on;

opts\_nq.Title.String = "B-7-19 Nyquist Diagram with Negative and Positive Feedback, Koike";

nyquistplot(G1,opts\_nq, 'blue-')

nyquistplot(G2,opts\_nq, 'red--')

legend('Negative-feedback', 'Positive-feedback')

axis equal; xlim([-8,8]);ylim([-8,8]);

saveas(fig,fullfile(fdir,"B-7-19\_nyquist2.png"));

### B-7-24

% Define transfer function

num = [0 25];

den = conv([1 0],[1 1]); den = conv(den,[1 10]);

G = tf(num,den);

% Find zeros, poles, and corner frequencies

zrs = roots(num);

pls = roots(den);

cornFreq = corner\_freq(num,den);

% Bode Plot

fig = figure("Renderer","painters");

opts\_bd.Title.String = "B-7-24 Bode Diagram, Koike";

bd = bodeplot(G,opts\_bd);

opt = getoptions(bd);

saveas(fig,fullfile(fdir,"B-7-24\_bode.png"));

% Phase Margin

[Wgc,phi] = gain\_crossover(num,den);

PM = 180 + phi;

% Gain Margin

[Wpc,Kg\_inv] = phase\_crossover(num,den);

GM = 1/abs(Kg\_inv);

GM\_dB = 20\*log10(1/abs(Kg\_inv)); % [dB]

% Validate with builtin function

[Gm\_v,Pm\_v,Wgc\_v,Wpc\_v] = margin(G);

fprintf("The actual values for relative stability analysis.");

fprintf("Gain Margin: %.4f at gain crossover frequency of %.4f",Gm\_v,Wgc\_v);

fprintf("Phase Margin: %.4f at phase crossover frequency of %.4f\n",Pm\_v,Wpc\_v);

### B-7-26

% From phase margin

syms w

eqn = w == (4 - w^2)\*tand(40);

Wgc = double(solve(eqn,w));

Wgc = Wgc(Wgc>0);

K = sqrt(Wgc^6 - 7\*Wgc^4 + 16\*Wgc^2);

% Define transfer function

num = [0 K];

den = conv([1 0],[1 1 4]);

G = tf(num,den);

% Gain margin

[Wpc,Kg\_inv] = phase\_crossover(num,den);

GM = 1/abs(Kg\_inv);

GM\_dB = 20\*log10(GM); % [dB]

### P3 Aircraft Example

% Define transfer function

num = [1.1057 -0.19];

den = [1 17.95 123.3 366.3 112.2 0];

G = tf(num,den);

pls = roots(den)

zrs = roots(num)

cornFreq = corner\_freq(num,den)

fig = figure("Renderer","painters");

opts\_bd.Title.String = "Aircraft Example Bode Diagram, Koike";

bodeplot(G,opts\_bd);

saveas(fig,fullfile(fdir,"P3\_bode.png"));

% Sample out points from Bode plot

res = bode\_sample\_points(G,[0.01,0.1,1,10,50]);

writetable(res,fullfile(fdir,"P3.xls"),"WriteMode","overwritesheet");

% Nyquist Plot

fig = figure("Renderer","painters");

opts\_nq.Title.String = "Aircraft Example Nyquist Diagram, Koike";

nyquistplot(G,opts\_nq);

xlim([-0.001 0.015])

saveas(fig,fullfile(fdir,"P3\_nyquist.png"));

### P4 Spacecraft Example

num = 0.036\*[1 25];

den = [1 0.04 1 0 0];

pls = roots(den);

zrs = roots(num);

cornFreq = corner\_freq(num,den);

G = tf(num,den);

fig = figure("Renderer","painters");

opts\_bd.Title.String = "Spacecraft Example Bode Diagram, Koike";

bodeplot(G,opts\_bd);

saveas(fig,fullfile(fdir,"P4\_bode.png"));

% Sample out points from Bode plot

res = bode\_sample\_points(G,[0.2,0.7,1,2,10,100,600]);

writetable(res,fullfile(fdir,"P4.xls"),"WriteMode","overwritesheet");

% Nyquist Plot

fig = figure("Renderer","painters");

opts\_nq.Title.String = "Aircraft Example Nyquist Diagram, Koike";

nyquistplot(G,opts\_nq);

saveas(fig,fullfile(fdir,"P4\_nyquist.png"));

function w\_i = corner\_freq(num,den)

%{

Function: corner\_freq()

Author: Tomoki Koike

Description: Computes the corner frequencies for a Bode Plot.

>>Inputs

num: the numerator of the open-loop transfer function

den: the denominator of the open-loop transfer function

Outputs<<

w\_i: the table with the corner frequencies for poles and zeros

%}

pls = roots(den);

zrs = roots(num);

cornP = unique(abs(pls));

cornZ = unique(abs(zrs));

if length(cornP) > length(cornZ)

cornZ = [cornZ; NaN([(length(cornP) - length(cornZ)), 1])];

else

cornP = [cornP; NaN([(length(cornZ) - length(cornP)), 1])];

end

w\_i = array2table([cornP, cornZ],"VariableNames",{'Poles','Zeros'});

end

function res\_T = bode\_sample\_points(sys,samp)

%{

Function: bode\_sample\_points()

Author: Tomoki Koike

Description: Finds magnitude and phase points corresponding to the

sample points provided as the user input.

>>Inputs

sys: the system/transfer function

samp: the sample frequency points

Outputs<<

res\_T: the table with all the results: points, frequencies, phase

angles in degrees, magnitudes in dB, and magnitudes.

%}

% Call the bode function to obtain required data points

omg = logspace(-2,3,1000000);

[mag,phase,w] = bode(sys,omg);

mag = mag(:); phase = phase(:);

% Find corresponding values for sample frequencies using 1D data

% interpolation

mag\_pt = interp1(w,mag,samp);

phase\_pt = interp1(w,phase,samp);

magdB\_pt = -20\*log10(mag\_pt);

% check if samp vector is row and if it is a row vector transpose to make it a

% column vector

if isrow(samp)

samp = samp.';

end

% Construct array with results

arr = [samp, phase\_pt.', magdB\_pt.', mag\_pt.'];

res\_T = array2table(arr,"VariableNames",{'Frequencies','Phase','MagdB','Mag'});

end

function [Wpc,res] = phase\_crossover(num,den)

%{

Function: phase\_crossover()

Author: Tomoki Koike

Description: Computes the intersection of the real axis and the

Nyquist plot.

>>Inputs

num: the numerator of the open-loop transfer function

den: the denominator of the open-loop transfer function

Outputs<<

Wpc: phase crossover frequency

res: intersections/real number of the phase crossover point

%}

% Get the length of each numerator and denominator

num\_len = length(num);

den\_len = length(den);

% Preset a array with the order of magnitudes (i.e. s^3, s^2, s^1, s^0)

% corresponding to the numerator and denominator

O\_num = (num\_len-1):-1:0;

O\_den = (den\_len-1):-1:0;

% Define a system equation to find phase crossover

w = sym('w');

assume(w,'real');

N = dot(num,(w\*(1j)).^O\_num);

D = dot(den,(w\*(1j)).^O\_den);

fprintf('The denominator factored out.\n'); disp(vpa(D,6));

NUM = N\*conj(D);

fprintf('The numerator combined with denominator.\n'); disp(vpa(NUM,6));

NUM\_im = imag(NUM);

fprintf('The imaginary part you need to equate to 0.\n'); disp(vpa(NUM\_im,6));

eqn = NUM\_im == 0;

Wpc = double(solve(eqn,w));

fprintf('Omegas that make the imaginary part equal 0 (phase crossover frequency).\n'); disp(Wpc);

G = N/D;

Wpc = Wpc(Wpc~=0 & Wpc>0); % Take out 0 and negative values

res = double(subs(G,w,Wpc));

end

function [Wgc,phi] = gain\_crossover(num,den)

%{

Function: gain\_crossover()

Author: Tomoki Koike

Description: Computes the intersection of the real axis and the

Nyquist plot.

>>Inputs

num: the numerator of the open-loop transfer function

den: the denominator of the open-loop transfer function

Outputs<<

res: gain crossover frequency

phi: angle corresponding to the gain crossover frequency (deg)

%}

% Get the length of each numerator and denominator

num\_len = length(num);

den\_len = length(den);

% Preset a array with the order of magnitudes (i.e. s^3, s^2, s^1, s^0)

% corresponding to the numerator and denominator

O\_num = (num\_len-1):-1:0;

O\_den = (den\_len-1):-1:0;

% Define a system equation to find the gain crossover frequency

w = sym('w');

assume(w,'real');

N = dot(num,(w\*(1j)).^O\_num);

D = dot(den,(w\*(1j)).^O\_den);

eqn = (abs(N))^2 == (abs(D))^2;

fprintf('Equation of |G(jw)| = 1.\n'); disp(eqn);

Wgc = double(solve(eqn,w));

% Filter result to be one value

Wgc = unique(abs(Wgc));

% Compute corresponding angle from the found frequency

G = tf(num,den);

s = Wgc\*1i;

phi = rad2deg(angle(evalfr(G,s)));

end