TMME50 Assignment II

David Wiman
20000120-8495

November 17, 2023

Instructions 2023 for reporting the computer assignments

The computer assignments are reported in writing, and submitted *printed on paper*. The assignments are performed individually. The use of ChatGPT or any similar system is not allowed. It is permissible to discuss the assignments and to show parts of solutions in that context, but *copying of Matlab code or sections of reports is not allowed.* Further, it is not allowed to possess copies of other students reports or Matlab code, either electronically or on paper, or to supply this to another student; this also means that you hand in and pick up your assignments yourself, not with the help of a friend. The reports shall contain:

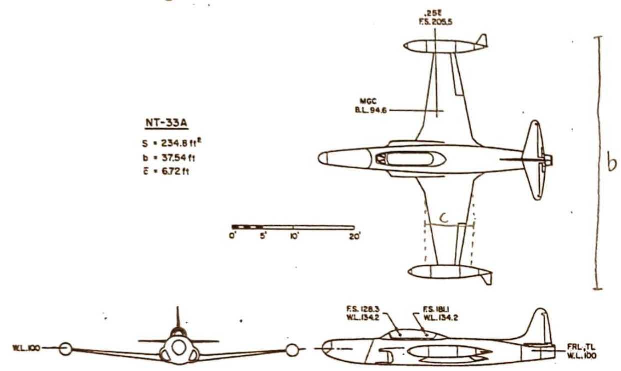
- *A copy of this page of instructions.*
- Name and complete (10 digits) civic registration number of the student (sometimes called p-number among exchange students).
- *Which aeroplane and which reference condition* that has been used. Specify the number of the column on the data sheet that has been used.
- Answers to all the questions appearing under the headings "Assignment I:a" etc. and all requested plots.
- A complete set of Matlab files for each computer assignment. Choose the most complete set, such as the one for part I:c in assignment I. In assignment II, also also include root loci and a graphic representation of your Simulink model for the final version of your model with all numerical values shown explicitly.
- The ODE system implemented in assignments I, III, IV and V must be given in the report in the order actually implemented and written in a *single* frame containing *all* the equations of the ODE and *nothing* else.

Further, note:

- With the exeption of flying qualities tables, illustrations from the lab-PM defining the problem and this page of instructions, no copying of text, figures, equations or code from another document is allowed (unless it is a document you have created yourself).
- It must be clear what data has been used in what way. Data is converted from American to SI units, and this should be done in a way that can be followed in detail either in the text of the report or in the Matlab files, so that misstakes can be found at a glance without making any calculations.
- Nothing written in Matlab syntax or pseudocode is allowed in the main text of the reports: your Matlab code is appended at the end of the report.
- The report must contain sufficient text and illustrations such that it is possible to understand without ever having seen the lab-PM.
- Use the simulation time given in the assignments. For a small number of datasets it is necessary to use a longer simulation time than 100 s in order to to see a full phugoid period, but the time should never be shorter than the time given and never longer than 400 s.

T-33 Shooting Star

David Wiman



	1	2	3	4	5	6	7	8
X_u s^{-1}	−0,0391	−0,00484	−0,0104	−0,0415	0,00477	−0,00735	−0,0511	−0,0355
X_α ft/s^2	18,58	35,37	25,12	−16,50	20,43	22,29	7,67	24,59
Z_u s^{-1}	−0,248	−0,153	−0,128	−0,162	−0,114	−0,107	−0,0703	−0,0766
Z_α ft/s^2	−213,41	−267,57	−773,31	−2807	−140,26	−712,5	−1400	−437,8
M_u $ft^{-1}s^{-1}$	0,000318	0,000603	0,000283	−0,00076	0,000114	−0,000183	−0,00151	−0,000183
M_α s^{-2}	−1,89	−1,81	−9,21	−33,7	−0,23	−9,95	−18,59	−5,42
$M_{\dot\alpha}$ s^{-1}	−0,35	−0,40	−0,63	0	−0,24	−0,31	−0,16	−0,06
M_q s^{-1}	−0,694	−0,806	−1,37	−2,80	−0,50	−0,981	−1,56	−0,535
X_{δ_e} ft/s^2	0,516	1,47	0,62	−2,65	1,88	0,50	−0,432	0,996
Z_{δ_e} ft/s^2	−13,4	−16,2	−44,4	−152	−11,3	−40,9	−82,4	−23,8
M_{δ_e} s^{-2}	−4,19	−5,83	−16,0	−52,7	−4,13	−14,2	−28,7	−8,28

h
V, u_0
m
I_x
I_y
I_z
I_{xz}
Q

surface, span, chord

$S = 234,8\ ft^2$, $b = 37,54\ ft$, $\bar{c} = 6,72\ ft$

	1	2	3	4	5	6	7	8
h ft	S/L	S/L	S/L	S/L	20000	20000	20000	40000
V(u_0) ft/s	228	270	447	782	311	570	778	629
m slug	367	426	426	426	426	426	426	426
I_x $slug \cdot ft^2$	12700	23801	23801	23801	23801	23801	23801	23801
I_y $slug \cdot ft^2$	20700	21101	21101	21101	21101	21101	21101	21101
I_z $slug \cdot ft^2$	32001	43802	43802	43802	43802	43802	43802	43802
I_{xz} $slug \cdot ft^2$	480	480	480	480	480	480	480	480
Q lb/ft^2	61,7	86,7	247	726	61,3	206	383	117

	1	2	3	4	5	6	7	8
Y_β ft/s^2	−28,4	−30,1	−81,0	−264	−21,6	−72,7	−144	−42,4
L_β s^{-2}	−5,49	−4,72	−8,02	−18,0	−4,06	−7,42	−9,89	−5,08
L_p s^{-1}	−2,03	−1,32	−2,15	−4,51	−0,82	−1,56	−2,23	−0,877
L_r s^{-1}	0,641	0,305	0,320	0,495	0,214	0,256	0,328	0,179
N_β s^{-2}	0,667	0,94	2,71	10,6	0,54	2,60	6,24	1,68
N_p s^{-1}	−0,116	−0,112	−0,0512	0,0118	−0,103	−0,0393	−0,0141	−0,0428
N_r s^{-1}	−0,207	−0,173	−0,291	−0,561	−0,104	−0,204	−0,318	−0,110
Y_{δ_r} ft/s^2	0,0295	0,0301	0,0503	0,102	0,0185	0,0363	0,0571	0,0195
L_{δ_r} s^{-2}	−0,0125	0,443	1,57	5,89	0,287	1,39	3,20	0,808
N_{δ_r} s^{-2}	−1,24	−1,25	−3,50	−12,6	−0,883	−3,21	−6,99	−1,92
L_{δ_a} s^{-2}	6,01	4,53	12,6	47,0	3,14	11,7	24,0	7,13
N_{δ_a} s^{-1}	0,0286	0,134	0,165	0,260	0,164	0,121	0,195	0,118

Background

In this assignment, we will study the longitudinal motion of an aeroplane, considering the short-period mode only.

To model some importent aspects of the aerodynamics of an aeroplane, consider the following transfer functions for the pitch angular velocity $\Delta\tilde{q}$ and the z-direction velocity $\Delta\tilde{\omega}$:

$$\Delta\tilde{q} = \Delta\dot{\tilde{\theta}} = \frac{b_1 s + b_0}{s^2 + 2\zeta_{sp}\omega_{n,sp}s + \omega_{n,sp}^2}\Delta\tilde{\delta}_e = G_q(s)\Delta\tilde{\delta}_e \tag{1}$$

$$\Delta\tilde{\omega} = \frac{d_1 s + d_0}{s^2 + 2\zeta_{sp}\omega_{n,sp}s + \omega_{n,sp}^2}\Delta\tilde{\delta}_e = G_\omega(s)\Delta\tilde{\delta}_e \tag{2}$$

where

$$\begin{aligned}
b1 &= M_{\delta_e} + M_{\dot{\omega}}Z_{\delta_e} \\
b0 &= M_\omega Z_{\delta_e} - M_{\delta_e}Z_\omega \\
d1 &= Z_{\delta_e} \\
d0 &= -M_q Z_{\delta_e} + u_0 M_{\delta_e} \\
\omega_{n,sp} &= \sqrt{Z_\omega M_q - M_\omega u_0} \\
\zeta_{sp} &= \frac{-1}{2\omega_{n,sp}}(M_q + M_{\dot{\omega}}u_0 + Z_\omega)
\end{aligned} \tag{3}$$

These are valid for small deviations from a trimmed flight condition with velocity u_0 in the x-direction.

A transfer function for the z-direction acceleration $\Delta\tilde{a}_z$ at a distance l from the center of mass along the x-axis can be calculated from:

$$\Delta\tilde{a}_z(l) = s\Delta\tilde{\omega} - u_0\Delta\tilde{q} - ls\Delta\tilde{q} = G_{a_z}(s;l)\Delta\tilde{\delta}_e \tag{4}$$

II:a

First, I write down the transfer function for $\Delta\tilde{a}_z$ at location l by using (1), (2) and (4).

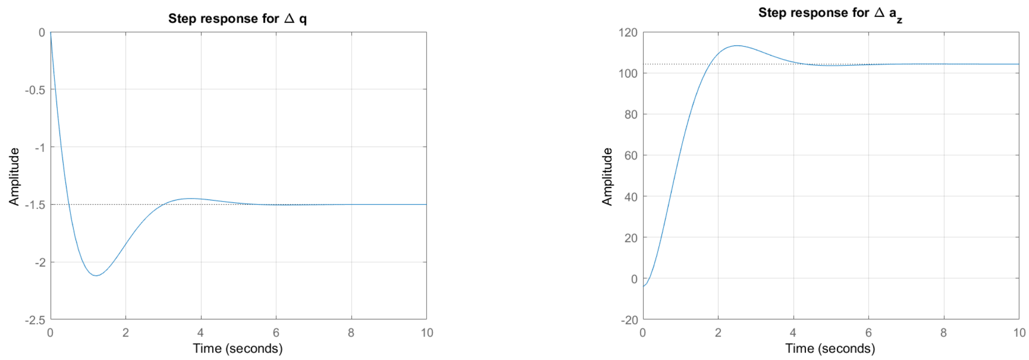
$$\Delta\tilde{a}_z = \frac{(d_1 s + d_0)s - (b_1 s + b_0)u_0 - (b_1 s + b_0)ls}{s^2 + 2\zeta_{sp}\omega_{n,sp}s + \omega_{n,sp}^2}\Delta\tilde{\delta}_e \tag{5}$$

I use my numerical values from a T-33 Shooting Star at sea level flight (column one on the data sheet) together with $l = 0$ and compute $G_q(s)$ and $G_{a_z}(s;l)$, I get

$$\begin{aligned}
G_q(s) &= \frac{-4.169s - 3.811}{s^2 + 1.98s + 2.54} \\
G_{a_z}(s) &= \frac{-4.084s^2 - 4.264s + 264.8}{s^2 + 1.98s + 2.54}
\end{aligned} \tag{6}$$

II:b

I then calculate the step responses for 10 seconds for Δq and Δa_z at the center of mass ($l = 0$). The results can be seen in Figure 1.



(a) Step response for Δq.

(b) Step response for Δa_z.

Figure 1

I also calculate the step response for 100 seconds for Δq. The result can be seen in Figure 2.

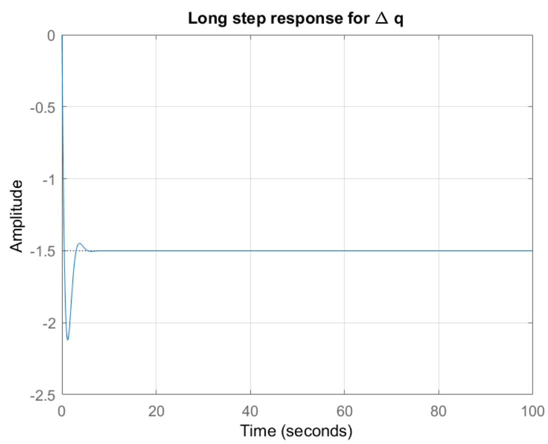
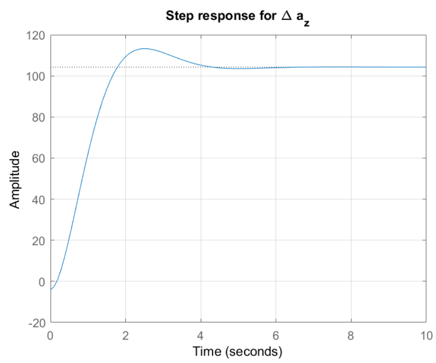
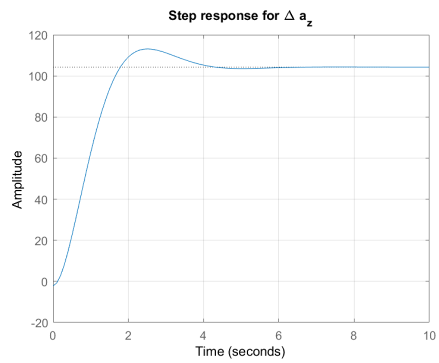


Figure 2: Long step response for Δq.

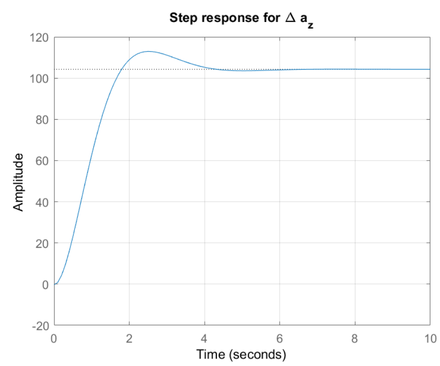
We can see that the normal acceleration Δa_z initially is in the wrong direction in Figure 1. Increasing l, I plot the step response again trying to find the center of rotation. These plots can be seen in Figure 3.



(a) Step response for Δa_z with $l = 0$.



(b) Step response for Δa_z with $l = 0.45$.



(c) Step response for Δa_z with $l = 0.9$.

Figure 3

We can see that with $l = 0.9$, the acceleration is no longer in the wrong direction.

II:c

The dynamical behavior of a linear system is by large determined by the poles of the transfer function of the system. I find these for my aeroplane short-period dynamics:

$$s^2 + 2\zeta_{sp}\omega_{n,sp}s + \omega_{n,sp}^2 = 0 \tag{7}$$

$$(s + \zeta_{sp}\omega_{n,sp})^2 - \zeta_{sp}^2\omega_{n,sp}^2 + \omega_{n,sp}^2 = 0 \tag{8}$$

$$(s + \zeta_{sp}\omega_{n,sp})^2 = \zeta_{sp}^2\omega_{n,sp}^2 - \omega_{n,sp}^2 \tag{9}$$

$$s + \zeta_{sp}\omega_{n,sp} = \pm\sqrt{\zeta_{sp}^2\omega_{n,sp}^2 - \omega_{n,sp}^2} \tag{10}$$

$$s = -\zeta_{sp}\omega_{n,sp} \pm \sqrt{\zeta_{sp}^2\omega_{n,sp}^2 - \omega_{n,sp}^2} \tag{11}$$

$$\tag{12}$$

Plugging in numerical values from our given aircraft gives us

$$\begin{aligned} s_1 &= -0.9900 + 1.2488i, \\ s_2 &= -0.9900 - 1.2488i. \end{aligned} \tag{13}$$

II:d

The previously mentioned poles are not necessarily ideal for aeroplane behavior. I specify a criterion for desirable dynamics in terms of the Control Anticipation Parameter (CAP). I select $CAP^{ideal} = \zeta_{sp}^{ideal} = 0.7$

to get level 1 flying qualities. Together with the values in (3) and u_0, i get the ideal poles

$$\begin{aligned} r_1^{ideal} &= -1.4902 + 1.5204i \\ r_2^{ideal} &= -1.4902 - 1.5204i \end{aligned} \tag{14}$$

using the formulae

$$\begin{aligned} CAP^{ideal} &= \frac{(\omega_{n,sp}^{ideal})^2 g T_{\theta_2}}{u_0} \\ T_{\theta_2} &= b_1/b_0 \\ r_{1,2}^{ideal} &= -\zeta_{sp}^{ideal}\omega_{n,sp}^{ideal} \pm i\omega_{n,sp}^{ideal}(\sqrt{1-(\zeta_{sp}^{ideal})^2}) \end{aligned} \tag{15}$$

II:e

We add a electromechanical feedback system to the aeroplane on the form in Figure 4.

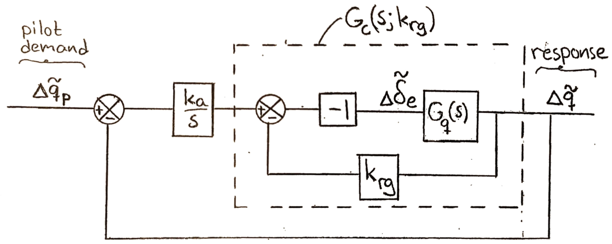


Figure 4: Diagram representing the aeroplane dynamics with a feedback system added.

The system has two variable gains, k_a and k_{rg}. I will try to select these such that the poles of the system are closer to the ideal poles.

First, I fix k_{rg} and name the inner loop in Figure 4 to $G_c(s;k_{rg})$. The system can now be seen in Figure 5.

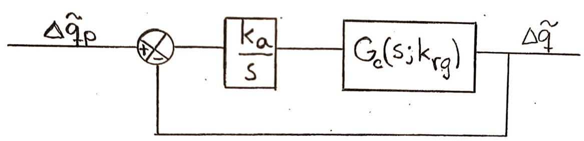


Figure 5: Diagram representing the aeroplane dynamics with a feedback system added, simplified

I now want to use the *rlocus* function in Matlab to get the root locus of the system with respect to k_a. *rlocus* assumes a system on the form seen to the right in Figure 6. My system has the form on the left in the same figure.

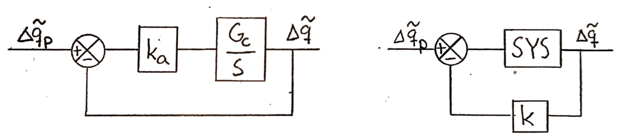


Figure 6: Diagram representation my system and the system assumed by *rlocus*.

The right system can be described by

$$\Delta\tilde{q} = \frac{\text{SYS}}{1 + k\text{SYS}}\Delta\tilde{q}_p \tag{16}$$

so passing $\frac{Gc}{s}$ to *rlocus* would result in plots describing

$$\Delta\tilde{q} = \frac{\frac{Gc}{s}}{1 + k\frac{Gc}{s}}\Delta\tilde{q}_p \tag{17}$$

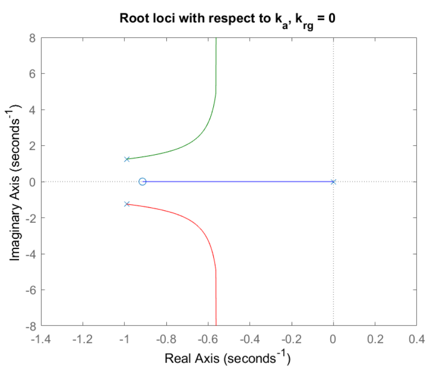
This is not the same as the left system, which can be described by

$$\Delta\tilde{q} = \frac{k_a\frac{G_c}{s}}{1 + k_a\frac{Gc}{s}}\Delta\tilde{q}_p \tag{18}$$

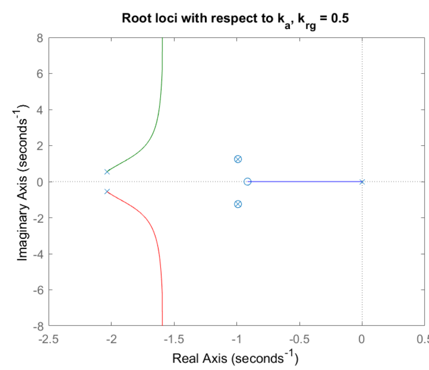
However, since the only difference is in the nominator, the poles of the system (roots of the denominator) are not effected. Therefore, we get the correct root locus.

II:f

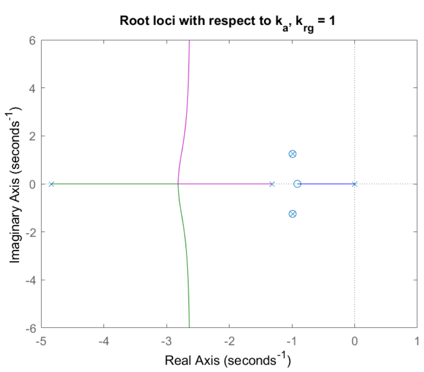
Using several values for k_{rg}, I compute G_c for each and use the *rlocus* function. The corresponding plots can be seen in Figure 7.



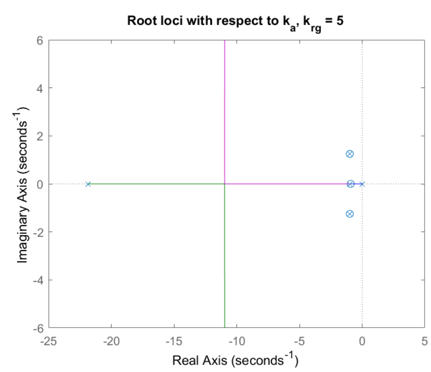
(a) Root locus for the system with respect to k_a. $k_{rg} = 0$.



(b) Root locus for the system with respect to k_a. $k_{rg} = 0.5$.



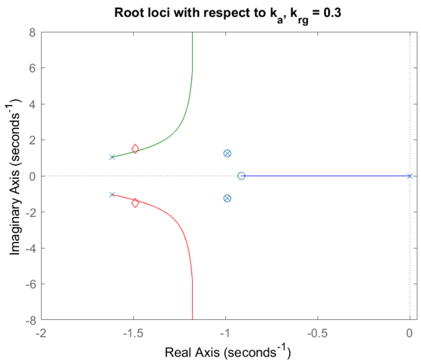
(c) Root locus for the system with respect to k_a. $k_{rg} = 1$.



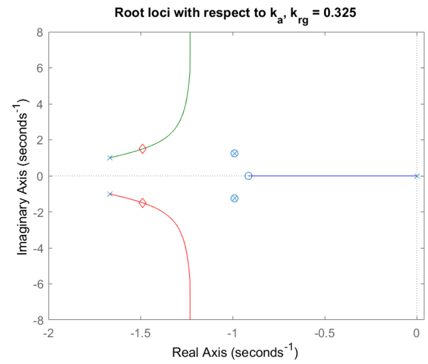
(d) Root locus for the system with respect to k_a. $k_{rg} = 5$.

Figure 7

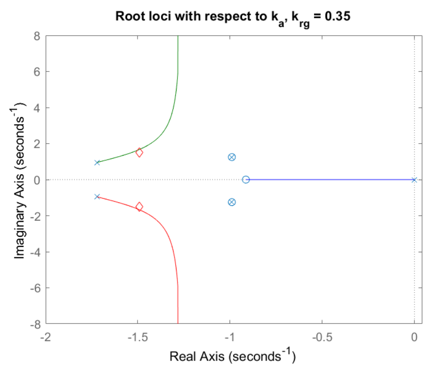
To be able to get poles close to the ideal poles, k_{rg} should be somewhere between 0 and 0.5. If I plot more root loci (with the ideal poles marked with red diamonds) with more accurate values for k_{rg} I get the result in Figure 8.



(a) Root locus for the system with respect to k_a. $k_{rg} = 0.3$.



(b) Root locus for the system with respect to k_a. $k_{rg} = 0.325$.



(c) Root locus for the system with respect to k_a. $k_{rg} = 0.35$.

Figure 8

I now select $k_{rg} = 0.325$ and $k_a = 0.417$ as these give me the poles closest to the ideal ones.

II:g

I now implement a model of the system in Simulink. This can be seen in Figure 9.

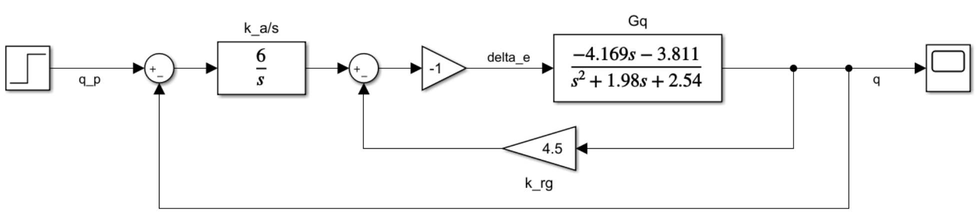


Figure 9: Simulink model of the system.

The values of $k_{rg} = 4.5$ and $k_a = 6$ have increased compared to the values in assignment II:f to decrease the rise time of the step response. The step response can be seen in Figure 10.

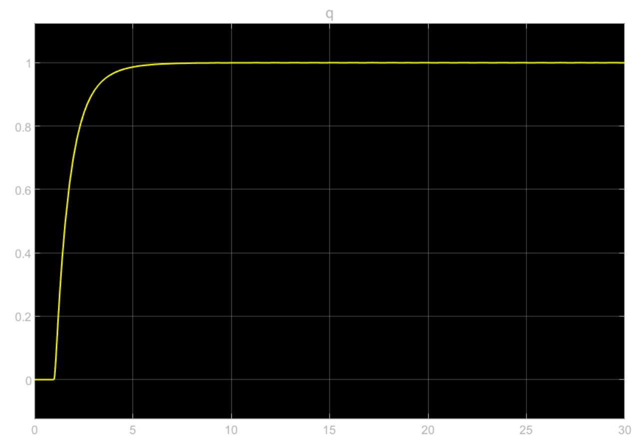


Figure 10: Step response for the system in Figure 9.

II:h

I now add a servo immediately before the aeroplane model $G_q(s)$ described by

$$G_{servo}(s) = \frac{\frac{1}{\tau}}{s + \frac{1}{\tau}} \tag{19}$$

This can be seen in Figure 11.

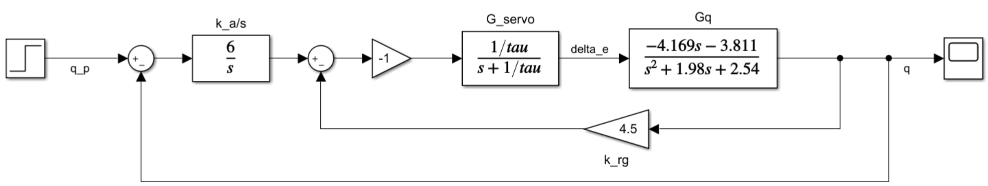
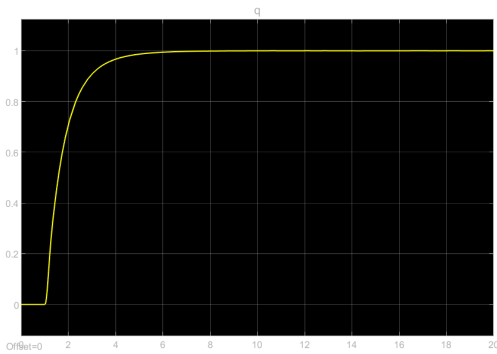
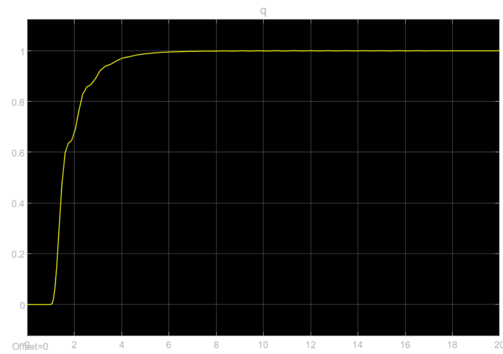


Figure 11: Simulink model of the system.

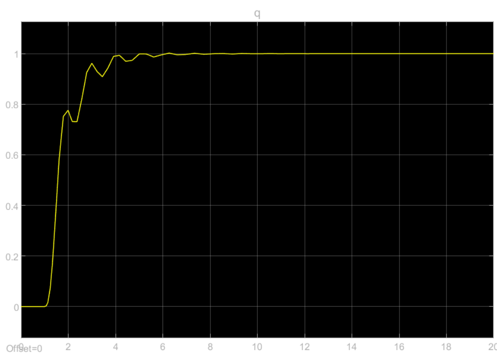
Now, I plot step responses using a few different values of τ. These can be seen in Figure 12.



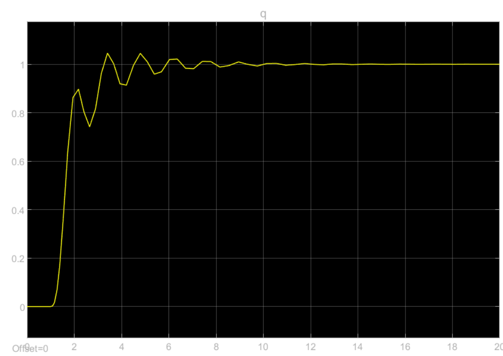
(a) Step response of the system with $\tau = 0.05$.



(b) Step response of the system with $\tau = 0.3$.



(c) Step response of the system with $\tau = 0.6$.



(d) Step response of the system with $\tau = 1$.

Figure 12

As is clear from Figure 12, it is possible to provoke oscillations. However, the oscillations are damped and the system remains stable.

1 Matlab code

1.1 Assignment II:a

```
% Set variables
run('T33_Shooting_Star_parameters.m');

% Create a Laplace variable
s = tf('s');

% Constants for the transfere functions
b1 = Mdelta_e + Mw_dot*Zdelta_e;
b0 = Mw*Zdelta_e - Mdelta_e*Zw;
d1 = Zdelta_e;
d0 = -Mq*Zdelta_e + v*Mdelta_e;
omega = sqrt(Zw*Mq - Mw*v);
zeta = -(Mq + Mw_dot*v + Zw) / (2*omega);
l = 0;

% Transfere function from elevator deflection to q
Gq = (b1*s + b0) / (s^2 + 2*zeta*omega*s + omega^2);

% Tansfere function from elevator deflection to w
Gw = (d1*s + d0) / (s^2 + 2*zeta*omega*s + omega^2);

% Tansfere function from elevator deflection to az
Gaz = minreal( s*Gw - v*Gq - l*s*Gq );
```

1.2 Assignment II:b

```
% Set variables
run('T33_Shooting_Star_parameters.m');

% Create a Laplace variable
s = tf('s');

% Constants for the transfere functions
b1 = Mdelta_e + Mw_dot*Zdelta_e;
b0 = Mw*Zdelta_e - Mdelta_e*Zw;
d1 = Zdelta_e;
d0 = -Mq*Zdelta_e + v*Mdelta_e;
omega = sqrt(Zw*Mq - Mw*v);
zeta = -(Mq + Mw_dot*v + Zw) / (2*omega);
l = 0.0; % Limit at 0.9

% Transfere function from elevator deflection to q
Gq = (b1*s + b0) / (s^2 + 2*zeta*omega*s + omega^2);

% Tansfere function from elevator deflection to w
Gw = (d1*s + d0) / (s^2 + 2*zeta*omega*s + omega^2);

% Tansfere function from elevator deflection to az
Gaz = minreal(s*Gw - v*Gq - l*s*Gq);
```

```

% Plot
figure(1)
step(Gq, 10);
title('Step response for \Delta q')
grid;

figure(2)
step(Gaz, 10);
title('Step response for \Delta a_z')
grid;

figure(3)
step(Gq, 100);
title('Long step response for \Delta q')
grid;
```

1.3 Assignment II:d

```
% Set variables
run('T33_Shooting_Star_parameters.m');

% Constants for the transfere functions
b1 = Mdelta_e + Mw_dot*Zdelta_e;
b0 = Mw*Zdelta_e - Mdelta_e*Zw;
d1 = Zdelta_e;
d0 = -Mq*Zdelta_e + v*Mdelta_e;
omega = sqrt(Zw*Mq - Mw*v);
zeta = -(Mq + Mw_dot*v + Zw) / (2*omega);
l = 0;

% Chosen CAP parameters
CAP = 0.7;
zeta_ideal = 0.7;

% Ideal poles
T_theta_2 = b1/b0;
omega_ideal = sqrt( (CAP*v)/(g*T_theta_2) );
r1 = -zeta_ideal*omega_ideal + 1i*omega_ideal*sqrt( 1 - zeta_ideal^2 );
r2 = -zeta_ideal*omega_ideal - 1i*omega_ideal*sqrt( 1 - zeta_ideal^2 );
```

1.4 Assignment II:e

```
% Set variables
run('T33_Shooting_Star_parameters.m');

% Create a Laplace variable
s = tf('s');

% Constants for the transfere functions
b1 = Mdelta_e + Mw_dot*Zdelta_e;
b0 = Mw*Zdelta_e - Mdelta_e*Zw;
```

```
d1 = Zdelta_e;
d0 = -Mq*Zdelta_e + v*Mdelta_e;
omega = sqrt(Zw*Mq - Mw*v);
zeta = -(Mq + Mw_dot*v + Zw) / (2*omega);
l = 0;

% Transfer function from elevator deflection to q
Gq = (b1*s + b0) / (s^2 + 2*zeta*omega*s + omega^2);

% Transfer function from elevator deflection to w
Gw = (d1*s + d0) / (s^2 + 2*zeta*omega*s + omega^2);

% Transfer function from elevator deflection to az
Gaz = s*Gw - v*Gq - l*s*Gq;

% Higher level transfer functions
k_rg = 1;
Gc = -Gq / (1 - k_rg*Gq); % Same as feedback(-Gq*k_rg, 1)
```

1.5 Assignment II:f

```
% Set variables
run('T33_Shooting_Star_parameters.m');

% Create a Laplace variable
s = tf('s');

% Constants for the transfere functions
b1 = Mdelta_e + Mw_dot*Zdelta_e;
b0 = Mw*Zdelta_e - Mdelta_e*Zw;
d1 = Zdelta_e;
d0 = -Mq*Zdelta_e + v*Mdelta_e;
omega = sqrt(Zw*Mq - Mw*v);
zeta = -(Mq + Mw_dot*v + Zw) / (2*omega);
l = 0;

% Transfere function from elevator deflection to q
Gq = (b1*s + b0) / (s^2 + 2*zeta*omega*s + omega^2);

% Tansfere function from elevator deflection to w
Gw = (d1*s + d0) / (s^2 + 2*zeta*omega*s + omega^2);

% Tansfere function from elevator deflection to az
Gaz = s*Gw - v*Gq - l*s*Gq;

% Higher level fransfere functions
k_rg = 0.325; % 0.325 gives good results
Gc = -Gq / (1 - k_rg*Gq); % Same as feedback(-Gq*k_rg, 1)

%  The poles on the root locus plot are denoted
%  by x and the zeros are denoted by o.
rlocus(Gc/s)
title('Root loci with respect to k_a, k_{rg} = 0.325')
```

```
hold on
plot(-1.4902, 1.5204, 'rd') % r1
plot(-1.4902, -1.5204, 'rd') % r2

% Store a good value
k_a = 0.417;
```

1.6 Unit conversion

```
% Constants
g = 9.81;

% Unit conversions
feet_to_meter = 0.3048;
feet2_to_m2 = 0.09290;
lb_to_kg = 0.4536;
slug_to_kg = 14.59;
slug_feet2_to_kg_m2 = 1.356;
lb_to_N = 4.448;

% Parameters in SI units
h = 100;
v = 228*feet_to_meter;
m = 367*slug_to_kg;
I_x = 12700*slug_feet2_to_kg_m2;
I_y = 20700*slug_feet2_to_kg_m2;
I_z = 32001*slug_feet2_to_kg_m2;
I_xz = 480*slug_feet2_to_kg_m2;
Q = 61.7*lb_to_N/feet2_to_m2;

Xu = -0.0391;
Xalpha = 18.58*feet_to_meter;
Xw = Xalpha/v;
Zu = -0.248;
Zalpha = -213.41*feet_to_meter;
Zw = Zalpha/v;
Mu = 0.000318/feet_to_meter;
Malpha = -1.89;
Mw = Malpha/v;
Malpha_dot = -0.35;
Mw_dot = Malpha_dot/v;
Mq = -0.694;
Xdelta_e = 0.516*feet_to_meter;
Zdelta_e = -13.4*feet_to_meter;
Mdelta_e = -4.19;

Ybeta = -28.4*feet_to_meter;
Lbeta = -5.49;
Lp = -2.03;
Lr = 0.641;
Nbeta = 0.667;
Np = -0.116;
Nr = -0.207;
```

```
Ydelta_r = 0.0295*feet_to_meter;
Ldelta_r = -0.0125;
Ndelta_r = -1.24;
Ldelta_a = 6.01;
Ndelta_a = 0.0286;

S = 234.8*feet2_to_m2;
b = 37.54*feet_to_meter;
c = 6.72*feet_to_meter;
```