TMME50 Assignment III

David Wiman
20000120-8495

December 8, 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:

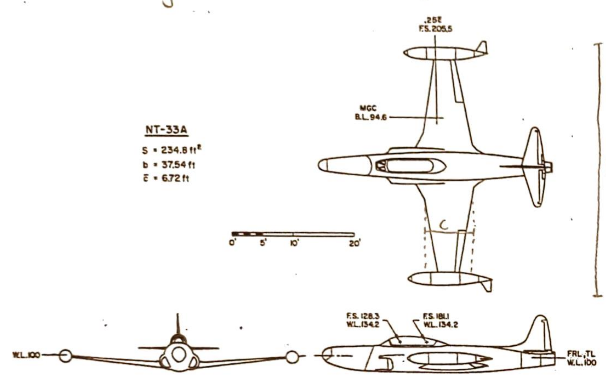
- *A copy of this page of instructions.*
- Name and complete (10 digits) civic registration number of the student (sometimes called p-number among exchange students).
- *Which aeroplane and which reference condition* that has been used. Specify the number of the column on the data sheet that has been 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	−8,95	−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

	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

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·ft²	12700	23801	23801	23801	23801	23801	23801	23801
I_y slug·ft²	20700	21101	21101	21101	21101	21101	21101	21101
I_z slug·ft²	32001	43802	43802	43802	43802	43802	43802	43802
I_{xz} slug·ft²	480	480	480	480	480	480	480	480
Q lb/ft²	61,7	86,7	247	726	61,3	206	383	117

Background

The force model used in previous assignments is extended by introducing contributions by $\Delta\delta_e$:

$$\begin{aligned} X &= mg\sin(\theta_0) + mX_u(u-u_0) + mX_w(w-w_0), \\ Z &= -mg\cos(\theta_0) + mZ_u(u-u_0) + mZ_w(w-w_0) + mZ_{\delta_e}\Delta\delta_e, \\ M &= I_{yy}M_w(w-w_0) + I_{yy}M_q(q-q_0) + I_{yy}M_{\delta_e}\Delta\delta_e. \end{aligned} \tag{1}$$

The data set for the tasks at hand were given in imperial units and must therefore be converted to SI units before use. Two of the values given were:

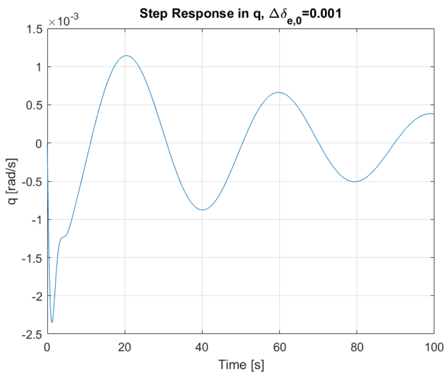
$$\begin{aligned} V &= 228 \text{ ft/s} \cdot 0.3048 = 69.4944 \text{ m/s}, \\ H &= 100 \text{ m}. \end{aligned} \tag{2}$$

I use data from a T-33 Shooting Star at sea level flight (column one on the data sheet).

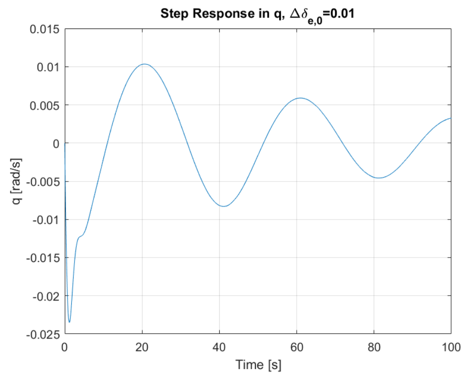
III:a

After implementing this new force model in Matlab, the system's response in q for a step in $\Delta\delta_e$ at time $t = 0$ was plotted. The result can be seen in Figure 1. The initial conditions (variable values from the data sheet) in (3) were used.

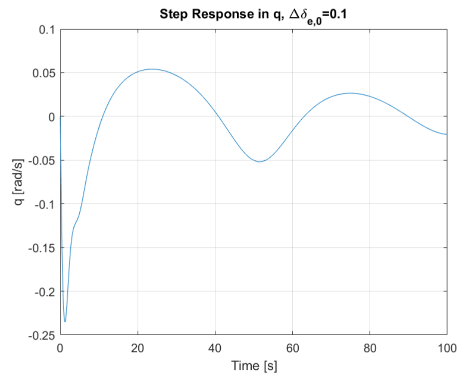
$$\begin{aligned} u_i &= V, \\ w_i &= 0, \\ q_i &= 0, \\ \theta_i &= 0, \\ x_f^i &= 0, \\ z_f^i &= H, \end{aligned} \tag{3}$$



(a) Response in q for step in $\Delta\delta_e$ with magnitude 0.001.



(b) Response in q for step in $\Delta\delta_e$ with magnitude 0.01.



(c) Response in q for step in $\Delta\delta_e$ with magnitude 0.1.

Figure 1

When I compare different step magnitudes, the overall shape of the response does not stay the same. There is an expected amplitude scaling, but there is also an unexpected change in shape.

This is because of the differential equation describing $\dot{q}$ contains both $\Delta\delta_e$ and w, which in turn depends on $\Delta\delta_e$. This makes q depend on both the first and second anti-derivative of $\Delta\delta_e$. See (4).

$$\begin{aligned} \dot{q} &= M_w(w - w_0) + M_q(q - q_0) + M_{\delta_e}\Delta\delta_e, \\ \dot{w} &= qu + g\cos(\theta) - g\cos(\theta_0) + Z_u(u - u_0) + Z_w(w - w_0) + Z_{\delta_e}\Delta\delta_e \end{aligned} \tag{4}$$

We do not see this behavior in assignment II where the differential equation for q does not depend on w, which can be seen in (5).

$$\Delta\tilde{q} = \frac{b_1 s + b_0}{s^2 + 2\zeta_{sp}\omega_{n,sp}s + \omega_{n,sp}^2}\Delta\tilde{\delta}_e \tag{5}$$

The non-linear differential equation gives more than just a scaling, it also affects the shape. The linear differential equation only scales the output.

When we compare the step response in q from this task with the similar task in assignment II (see Figure 2), we can see that the first few seconds look similar with a shape spike downwards, but after some time, they differ wildly. The model used in this task begins to oscillate instead of flattening out. This is because in assignment II, an approximation has been introduced in the model such that it is only valid for the short period motion. The model used in this task is also valid for the phugoid mode. Since the short period motion occurs right at the start of the simulation, the two different models are similar there, but not anywhere else.

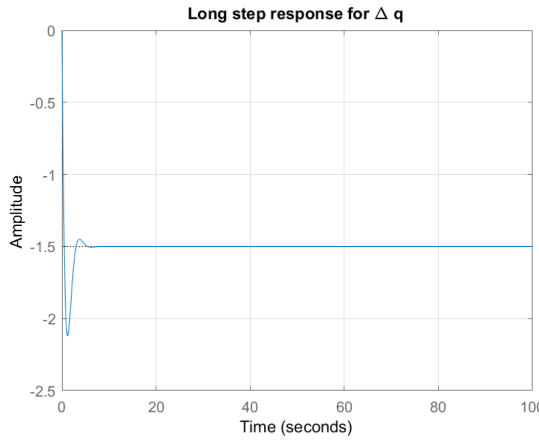


Figure 2: Step response in q for a step in $\Delta\delta_e$ from assignment II.

III:b

I will now introduce a regulator and servo to control the elevator.

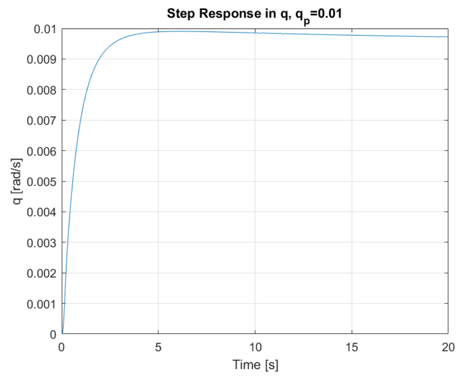
The elevator deflection $\Delta\delta_e$ will depend on a reference value q_p and the current response q as well as control parameters k_a and k_{rg}. The set of equations can be seen in (6).

$$\begin{aligned} \dot{e} &= k_a(q_p - q) \\ \Delta\delta_{in} &= -(e - k_{rg}q) \\ \Delta\dot{\delta}_e + \frac{\Delta\delta_e}{\tau} &= \frac{\Delta\delta_{in}}{\tau} \end{aligned} \tag{6}$$

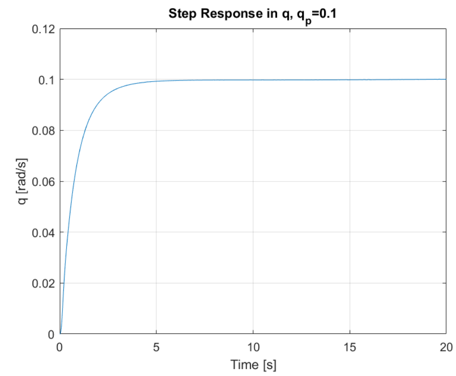
If $\Delta\delta_{in}$ is eliminated between the last two equations, we obtain, together with the equations of motion from previous assignments, the equations seen in (7).

$$\begin{aligned} \dot{u} &= -qw - g\sin(\theta) + \frac{X}{m}, \\ \dot{w} &= qu + g\cos(\theta) + \frac{Z}{m}, \\ \dot{q} &= \frac{M}{I_{yy}}, \\ \dot{\theta} &= q, \\ \dot{x}_f &= u\cos(\theta) + w\sin(\theta), \\ \dot{z}_f &= u\sin(\theta) - w\cos(\theta), \\ \dot{e} &= k_a(q_p - q), \\ \Delta\dot{\delta}_e &= -\frac{\Delta\delta_e}{\tau} - \frac{e - k_{rg}q}{\tau}. \end{aligned} \tag{7}$$

With these equations implemented in Matlab and using the values $k_a = 6$, $k_{rg} = 4.5$ and $\tau = 0.05$ computed in previous assignments and the assumption that e and $\Delta\delta_e$ are both initially zero, I plot the step response in q for steps in q_p. The result can be seen in Figure 3.



(a) Response in q for step in q_p with magnitude 0.01.



(b) Response in q for step in q_p with magnitude 0.1.

Figure 3

Compared to assignment II:h, there is no difference in how well q follows q_p. Both are very good.

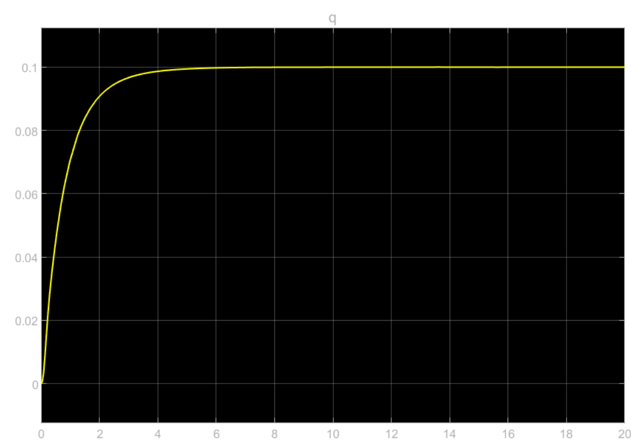
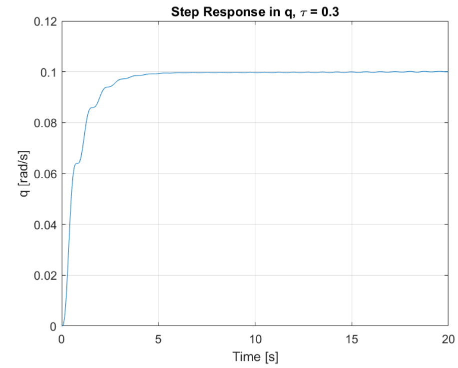
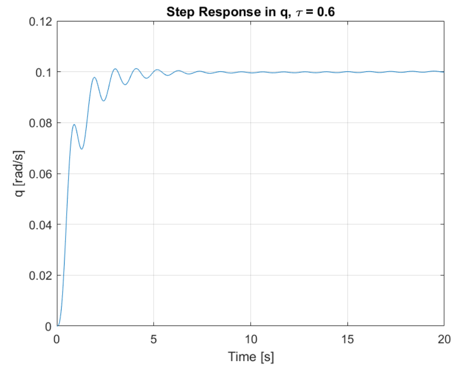


Figure 4: Response in q for step in q_p with magnitude 0.1 and $\tau = 0.05$ from assignment II:h.

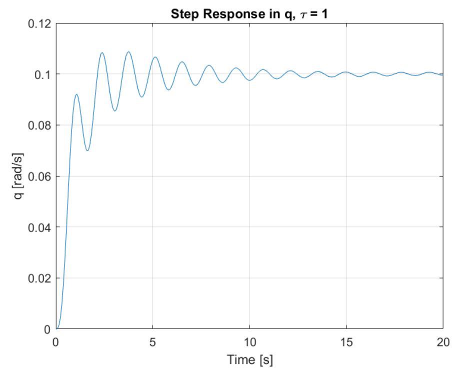
If I fix the step magnitude of $q_p = 0.1$ and instead vary τ, I see oscillations but no instability. This is similar to the result from assignment II. The result can be seen in Figure 5.



(a) Response in q for step in q_p with magnitude 0.1, $\tau = 0.3$.



(b) Response in q for step in q_p with magnitude 0.1, $\tau = 0.6$.



(c) Response in q for step in q_p with magnitude 0.1, $\tau = 1$.

Figure 5

1 Matlab code

1.1 Assignment III:a

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

% Reference state around which linearization happens
reference_state = [v, 0, 0, 0, 0, h];

% Force linearization parameters, [Xu, Xw, Zu, Zw, Mw, Mq]
force_lin_param = [Xu, Xw, Zu, Zw, Mw, Mq];

% Set ODE options
initial_conditions = [v, 0, 0, 0, 0, h];
time_interval = [0, 100];

% Set step magnitude
delta_e = 0.1;

% Compute ODE
[time, state] = ode45(@(t, state) state_propagation(t, state, m, I_y, delta_e,
    Zdelta_e, Mdelta_e, reference_state, force_lin_param), time_interval,
    initial_conditions);

% Extract all states for plotting
u = state(:,1);
w = state(:,2);
q = state(:,3);
theta = state(:,4);
x_f = state(:,5);
z_f = state(:,6);

% Plot
figure(1)
plot(time, q);
title('Step Response in q, \Delta\delta_{e,0}=0.1')
xlabel('Time [s]')
ylabel('q [rad/s]')
grid
```

1.2 Assignment III:a, system propagation

```
% Function to propagate the state
function next_state = state_propagation(t, state, m, I_yy, delta_e, Zdelta_e,
    Mdelta_e, reference_state, force_lin_param)
    % Rename state components for readability
    u = state(1);
    w = state(2);
    q = state(3);
    theta = state(4);
    x_f = state(5);
    z_f = state(6);
```

```

    % Extract reference state
    u0 = reference_state(1);
    w0 = reference_state(2);
    q0 = reference_state(3);
    theta_0 = reference_state(4);
    x_f_0 = reference_state(5);
    z_f_0 = reference_state(6);

    % Extract force linearization parameters
    Xu = force_lin_param(1);
    Xw = force_lin_param(2);
    Zu = force_lin_param(3);
    Zw = force_lin_param(4);
    Mw = force_lin_param(5);
    Mq = force_lin_param(6);

    % Constants
    g = 9.81;

    % Forces
    X = m*g*sin(theta_0) + m*Xu*(u - u0) + m*Xw*(w-w0);
    Z = -m*g*cos(theta_0) + m*Zu*(u - u0) + m*Zw*(w - w0) + m*Zdelta_e*delta_e
        ;
    M = I_yy*Mw*(w - w0) + I_yy*Mq*(q - q0) + I_yy*Mdelta_e*delta_e;

    % State dynamics
    next_state(1) = -q*w - g*sin(theta) + X/m;
    next_state(2) = q*u + g*cos(theta) + Z/m;
    next_state(3) = M/I_yy;
    next_state(4) = q;
    next_state(5) = u*cos(theta) + w*sin(theta);
    next_state(6) = u*sin(theta) - w*cos(theta);

    % Transpose to get row vector
    next_state = next_state';
end
```

1.3 Assignment III:b

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

% Reference state around which linearization happens
reference_state = [v, 0, 0, 0, 0, h];

% Force linearization parameters, [Xu, Xw, Zu, Zw, Mw, Mq]
force_lin_param = [Xu, Xw, Zu, Zw, Mw, Mq];

% Set ODE options
initial_conditions = [v, 0, 0, 0, 0, h, 0, 0];
time_interval = [0, 20];

```

```
% Set step magnitude
q_p = 0.1;

% Control parameters
k_a = 6;
k_rg = 4.5;
tau = 0.05;

% Compute ODE
[time, state] = ode45(@(t, state) state_propagation_control(t, state, m, I_y,
    q_p, k_a, k_rg, tau, Zdelta_e, Mdelta_e, reference_state, force_lin_param)
    , time_interval, initial_conditions);

% Extract all states for plotting
u = state(:,1);
w = state(:,2);
q = state(:,3);
theta = state(:,4);
x_f = state(:,5);
z_f = state(:,6);
e = state(:,7);
delta_e = state(:,8);

% Plot
figure(1)
plot(time, q);
title('Step Response in q, \tau = 0.05')
xlabel('Time [s]')
ylabel('q [rad/s]')
grid
```

1.4 Assignment III:b, system propagation

```
% Function to propagate the state
function next_state = state_propagation_control(t, state, m, I_yy, q_p, k_a,
    k_rg, tau, Zdelta_e, Mdelta_e, reference_state, force_lin_param)
    % Rename state components for readability
    u = state(1);
    w = state(2);
    q = state(3);
    theta = state(4);
    x_f = state(5);
    z_f = state(6);
    e = state(7);
    delta_e = state(8);

    % Extract reference state
    u0 = reference_state(1);
    w0 = reference_state(2);
    q0 = reference_state(3);
    theta_0 = reference_state(4);
    x_f_0 = reference_state(5);
    z_f_0 = reference_state(6);
```

```

    % Extract force linearization parameters
    Xu = force_lin_param(1);
    Xw = force_lin_param(2);
    Zu = force_lin_param(3);
    Zw = force_lin_param(4);
    Mw = force_lin_param(5);
    Mq = force_lin_param(6);

    % Constants
    g = 9.81;

    % Forces
    X = m*g*sin(theta_0) + m*Xu*(u - u0) + m*Xw*(w-w0);
    Z = -m*g*cos(theta_0) + m*Zu*(u - u0) + m*Zw*(w - w0) + m*Zdelta_e*delta_e
        ;
    M = I_yy*Mw*(w - w0) + I_yy*Mq*(q - q0) + I_yy*Mdelta_e*delta_e;

    % State dynamics
    next_state(1) = -q*w - g*sin(theta) + X/m;
    next_state(2) = q*u + g*cos(theta) + Z/m;
    next_state(3) = M/I_yy;
    next_state(4) = q;
    next_state(5) = u*cos(theta) + w*sin(theta);
    next_state(6) = u*sin(theta) - w*cos(theta);
    next_state(7) = k_a*(q_p - q);
    next_state(8) = -delta_e/tau - (e - k_rg*q)/tau;

    % Transpose to get row vector
    next_state = next_state';
end
```

1.5 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;
```