Table of Contents


```
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  ASEN 3128: Homework 6
%  Author: Caleb Bristol
%  Date: 26 April, 2022
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%


clear
close all;
clc
```

Problem 3

Part a)

```
   % Given
   DD = [-1.610e04 -3.062e05 2.131e05;0 -1.076e07 -1.330e06;0
9.925e06 -8.934e06];

   C_y_beta = -0.8771;
   C_l_beta = -0.2797;
   C_n_beta = 0.1946;
   C_y_p = 0;
   C_l_p = -0.3295;
   C_n_p = -0.04073;
   C_y_r = 0;
   C_l_r = 0.304;
   C_n_r = -0.2737;

   W = 2.83176e06;
   S = 511.0;
```

```
c = 8.324;
b = 59.64;
I_x = 0.247e08;
I_y = 0.449e08;
I_z = 0.673e08;
I_zx = -0.212e07;
u_0 = 235.9;
theta_0 = 0;
rho = 0.3045;
C_L_0 = 0.654;
C_D_0 = 0;

% Dimensionalize
Y_v = 0.5*rho*u_0*S*C_y_beta;
L_v = 0.5*rho*u_0*b*S*C_l_beta;
N_v = 0.5*rho*u_0*b*S*C_n_beta;
Y_p = 0.25*rho*u_0*b*S*C_y_p;
L_p = 0.25*rho*u_0*b^2*S*C_l_p;
N_p = 0.25*rho*u_0*b^2*S*C_n_p;
Y_r = 0.25*rho*u_0*b*S*C_y_r;
L_r = 0.25*rho*u_0*b^2*S*C_l_r;
N_r = 0.25*rho*u_0*b^2*S*C_n_r;

DD_ = [Y_v L_v N_v;Y_p L_p N_p;Y_r L_r N_r];

% Display Results
fprintf('Problem 3: \n \n')
fprintf('Part a) \n')
fprintf('Table 6.7: \n')
disp(DD)
fprintf('Dimensionalized Derivatives from Table 6.6: \n')
disp(DD_)
```

```
Problem 3:

Part a)
Table 6.7:
      -16100     -306200      213100
           0   -10760000    -1330000
           0     9925000    -8934000

Dimensionalized Derivatives from Table 6.6:
   1.0e+07 *

   -0.0016   -0.0306    0.0213
         0   -1.0755   -0.1329
         0    0.9923   -0.8934
```

Part b)

```
% i)
delta_p = 0.05; %[rad/s]
```

```
L_i = L_p * delta_p;

% ii)
delta_r = -0.05; %[rad/s]

N_ii = N_r * delta_r;

% iii)
delta_r = 0.01; %[rad/s]

L_iii = L_r * delta_r;

% iv)
delta_p = -0.7; %[rad/s]

N_iv = N_p * delta_p;

% v)
delta_p = 0.15; %[rad/s]
delta_v = 2.04; %[m/s]

Y_v = Y_p * delta_p + Y_v * delta_v;

% vi)
delta_v = -1.3; %[m/s]
delta_p = 0.5; %[rad/s]
delta_r = 0.37; %[rad/s]

N_vi = N_v * delta_v + N_p * delta_p + N_r * delta_r;

% Display Results
fprintf('Part b) \n')
fprintf('i) \n')
fprintf('Change in L due to perturbations [Nm]: \n')
disp(L_i)
fprintf('ii) \n')
fprintf('Change in N due to perturbations [Nm]: \n')
disp(N_ii)
fprintf('iii) \n')
fprintf('Change in L due to perturbations [Nm]: \n')
disp(L_iii)
fprintf('iv) \n')
fprintf('Change in N due to perturbations [Nm]: \n')
disp(N_iv)
fprintf('v) \n')
fprintf('Change in Y due to perturbations [N]: \n')
disp(Y_v)
fprintf('vi) \n')
fprintf('Change in N due to perturbations [Nm]: \n')
disp(N_vi)
fprintf('\n')
```

Part b)

```
i)
Change in L due to perturbations [Nm]:
  -5.3775e+05

ii)
Change in N due to perturbations [Nm]:
   4.4668e+05

iii)
Change in L due to perturbations [Nm]:
   9.9226e+04

iv)
Change in N due to perturbations [Nm]:
   9.3060e+05

v)
Change in Y due to perturbations [N]:
  -3.2839e+04

vi)
Change in N due to perturbations [Nm]:
  -4.2470e+06
```

Problem 4

Given

```
A_lat = [-0.0869 0 -0.825 32.175;-0.0054 -1.184 0.3350 0;0.0026 -0.0210 -0.2280 0;0 1.0000 0 0];
V = 825; %[ft/s]
h = 33000; %[ft]
theta_0 = 0; %[rad]
g = A_lat(1,4); %[ft/s^2]
```

Part a)

```
% Calculate Eigenvalues/Eigenvectors
[eigenvec,eigen] = eig(A_lat);

% Identify Dutch Roll Mode (2,2) and (3,3)
w_n = sqrt(real(eigen(2,2))^2 + imag(eigen(2,2))^2);
zeta = real(eigen(2,2)) / w_n;

% Identify Roll Mode (1,1)
t_roll = log(0.5) / real(eigen(1,1));

% Identify Spiral Mode (4,4)
t_spiral = log(0.5) / real(eigen(4,4));
```

Part b)

```
% Pull Variables from Matrix
N_r_ = A_lat(3,3);
L_v_ = A_lat(2,1);
N_v_ = A_lat(3,1);
L_r_ = A_lat(2,3);

% Compute Approximation
lambda_spi = (N_r_ * L_v_ - N_v_ * L_r_) / (L_v_);

t_spi_ = log(0.5) / lambda_spi;
```

Part c)

```
% Pull Variables from Matrix
L_p_ = A_lat(2,2);

% Compute Approximation
lambda_roll = L_p_;

t_roll_ = log(0.5) / lambda_roll;
```

Part d)

```
% Pull Variables from Matrix
N_p_ = A_lat(3,2);

% Compute Approximation
E = g * (N_r_ * L_v_ - N_v_ * L_r_);
D = V * (L_v_ * N_p_ - L_p_ * N_v_) - g * L_v_;
C = V * N_v_;

lambda_spi_comb = (-D + sqrt(D^2 - 4*C*E)) / (2 * C);
lambda_roll_comb = (-D - sqrt(D^2 - 4*C*E)) / (2 * C);

t_spi_comb = log(0.5) / lambda_spi_comb;
t_roll_comb = log(0.5) / lambda_roll_comb;
```

Part e)

```
% Pull Variables from Matrix
Y_v_ = A_lat(1,1);

% Compute Approximation
w_n_ = sqrt(Y_v_ * N_r_ + V * N_v_);
zeta_ = (Y_v_ + N_r_) / (2 * w_n_);
```

Display Results

```
fprintf('Problem 2: \n \n')
```

```
% Part a)
fprintf('Part a) \n')
fprintf('Eigenvectors: \n')
disp(eigenvec)
fprintf('Eigenvalues: \n')
disp(diag(eigen))
fprintf('Dutch Roll Mode: \n')
fprintf('Natural Frequency: \n')
disp(w_n)
fprintf('Damping Coefficient: \n')
disp(zeta)
fprintf('Spiral Mode Time to Half [s]: \n')
disp(t_spiral)
fprintf('Roll Mode Time to Half [s]: \n')
disp(t_roll)

% Part b)
fprintf('Part b) \n')
fprintf('Spiral Mode Approximation Time to Half [S]: \n')
disp(t_spi_)

% Part c)
fprintf('Part c) \n')
fprintf('Roll Mode Approximation Time to Half [s]: \n')
disp(t_roll_)

% Part d)
fprintf('Part d) \n')
fprintf('Combined Spiral and Roll Mode: \n')
fprintf('Combined Spiral Mode Approx Time to Half [s]: \n')
disp(t_spi_comb)
fprintf('Combined Roll Mode Approx Time to Half [s]: \n')
disp(t_roll_comb)

% Part e)
fprintf('Part e) \n')
fprintf('Dutch Roll Approximation: \n')
fprintf('Natural Frequency: \n')
disp(w_n_)
fprintf('Damping Coefficient: \n')
disp(zeta_)
```

```
Problem 2:

Part a)
Eigenvectors:
   0.9981 + 0.0000i   0.9999 + 0.0000i   0.9999 + 0.0000i   0.9999 +
 0.0000i
   0.0492 + 0.0000i  -0.0042 - 0.0005i  -0.0042 + 0.0005i  -0.0001 +
 0.0000i
  -0.0015 + 0.0000i   0.0027 - 0.0061i   0.0027 + 0.0061i   0.0159 +
 0.0000i
```

```
  -0.0378 + 0.0000i   0.0007 + 0.0112i   0.0007 - 0.0112i   0.0011 +
 0.0000i

Eigenvalues:
  -1.3034 + 0.0000i
  -0.0657 + 0.3664i
  -0.0657 - 0.3664i
  -0.0642 + 0.0000i

Dutch Roll Mode:
Natural Frequency:
    0.3722

Damping Coefficient:
   -0.1765

Spiral Mode Time to Half [s]:
   10.7995

Roll Mode Time to Half [s]:
    0.5318

Part b)
Spiral Mode Approximation Time to Half [S]:
   10.3914

Part c)
Roll Mode Approximation Time to Half [s]:
    0.5854

Part d)
Combined Spiral and Roll Mode:
Combined Spiral Mode Approx Time to Half [s]:
  167.3500

Combined Roll Mode Approx Time to Half [s]:
    0.5314

Part e)
Dutch Roll Approximation:
Natural Frequency:
    1.4713

Damping Coefficient:
   -0.1070
```

Comparisons Between Models

```
% Part b)
%
% The spiral mode approximation is incredibly close to the actual
```

```
   % calculated value, being off only by a couple of tenths of a
second.
   % Considering this error has been propagated through the
calculation of
   % time to half, the eigenvalue is only off by a couple hundredths,
and
   % remains on the same order of magnitude as the real value.

   % Part c)
   %
   % Much like the spiral mode approximation, the roll mode
approximation
   % is very accurate. It is only off by a couple hundredths of a
second,
   % though its error is very close to the spiral mode considering
that
   % while its error may be an order of magnitude less, so is its
value.

   % Part d)
   %
   % This approximation was somewhat close with the roll mode,
actually
   % being even closer to the original value than the individual roll
mode
   % approximation; however, the spiral mode was significantly off.
The
   % time to half provided by this approximation was off by an order
of
   % magnitude, and doesn't accurately reflect the actual behavior at
all.

   % Part e)
   %
   % This approximation was really far off with the natural
frequency,
   % being an order of magnitude off of the original value, though
   % strangely enough the damping ratio came out to be a somewhat
similar
   % value. Considering the fact that the damping ratio must be
between
   % zero and one for oscillations to occur, however, means this
relation
   % could very easily be more of a coincidence based on the
constraints
   % of the problem more than a gauge of the accuracy of this
   % approximation. Considering how far off the natural frequency was
from
   % the original value, I wouldn't consider this approximation
accurate
   % at all.
```

Published with MATLAB® R2020a