Indian Institute of Technology Bombay

AE 305 Flight Mechanics II, 2024

Report On

Course Project

F4 Phantom II Aircraft Mechanics

Disha Gupta	210010025
Ameya Marakarkandy	21D180003
Shubhranil Chatterjee	210100144
Rohan Suresh Mekal	22B2106

Contents

List of Figures

1 Introduction

The McDonnell Douglas F-4 Phantom II is a 4th gen American tandem two-seat, twin-engine, all-weather, long-range supersonic jet interceptor and fighter-bomber originally developed by McDonnell Aircraft for the United States Navy. Proving highly adaptable, it entered service with the Navy in 1961 before it was adopted by the United States Marine Corps and the United States Air Force, and by the mid-1960s it had become a major part of their air arms. Phantom production ran from 1958 to 1981 with a total of 5,195 aircraft built, making it the most-produced American supersonic military aircraft in history, and cementing its position as a signature combat aircraft of the Cold War.

Figure 1: A U.S. Air Force QF-4E

Being a fighter jet many of the assumptions that we will be using, like the Small-disturbance theory would break down as it does not apply to spinning motion, stall recovery, or any other application that involves rapid maneuvers or large-amplitude oscillations that are involved in a normal mission profile of a fighter jet. Also, it has 9 hardpoints and it is intended to be used even with assymetry so the assumption of symmetry about $i^b - k^b$ plane that we generally assume will also not hold.

In this project, we have created a simulation model of the F4 Phantom Aircraft using the data provided for cruise conditions using MATLAB and Simulink. This non-linear model is then trimmed at the cruise condition and linearized about the trim. The linearized model is then converted to lateral and longitudinal state space models using similarity transformation. Lateral and Longitudinal transfer functions are then found using these state space models.

2 Aircraft Specifications

2.1 General Characteristics

- Crew: 2
- Length: 63 ft 0 in (19.2 m)
- Wingspan: 38 ft 5 in (11.7 m)
- Width: 27 ft 7 in (8.4 m) wing folded
- Height: 16 ft 5 in (5 m)
- Wing area: 530 sq ft (49.2 m^2)
- Aspect ratio: 2.77
- Airfoil: NACA 0006.4–64 root, NACA 0003-64 tip
- Empty weight: 30,328 lb (13,757 kg)
- Gross weight: 41,500 lb (18,824 kg)
- Max takeoff weight: 61,795 lb (28,030 kg)
- Maximum landing weight: 36,831 lb (16,706 kg)
- Powerplant: 2 × General Electric J79-GE-17A after-burning turbojet engines, 11,905 lbf (52.96 kN) thrust each dry, 17,845 lbf (79.38 kN) with afterburner

2.2 Performance

- Maximum speed: 1,280 kn (1,470 mph, 2,370 km/h) at 40,000 ft (12,000 m)
- Maximum speed: Mach 2.23
- Cruise speed: 510 kn (580 mph, 940 km/h)
- Combat range: 370 nmi (420 mi, 680 km)
- Ferry range: 1,457 nmi (1,677 mi, 2,699 km)
- Service ceiling: 60,000 ft (18,000 m)
- Rate of climb: 41,300 ft/min (210 m/s)
- Lift-to-drag: 8.58
- Wing loading: 78 lb/sq ft (380 kg/m^2)
- Thrust/weight: 0.86 at loaded weight, 0.58 at MTOW
- Takeoff roll: 4,490 ft (1,370 m) at 53,814 lb (24,410 kg)
- Landing roll: 3,680 ft (1,120 m) at 36,831 lb (16,706 kg)

3 Aircraft Parameters

The aircraft parameters were obtained from Roskam[1] for F4 Phantom II Aircraft at subsonic cruise flying conditions.

```
% F4 Phantom II Aircraft Parameters
% Author: Ameya Marakarkandy
% Last Updated: 04-05-24


% Physical parameters of airframe
AC.gravity = 9.81;
AC.mass = 17690.102;

% Inertia Matrix
AC.Jx    = 33895.449047;
AC.Jy    = 165680.95494;
AC.Jz    = 189543.35107;
AC.Jxz   = 2982.7995162;
AC.Jxy = 0;
AC.Jyz = 0;
AC.J = [AC.Jx,AC.Jxy,AC.Jxz;...
        AC.Jxy,AC.Jy,AC.Jyz;...
        AC.Jxz,AC.Jyz,AC.Jz];

% Wing parameters
AC.S              = 49.2386;
AC.b              = 11.79576;
AC.c              = 4.8768;
AC.e              = 0.9;
AC.AR             = AC.b^2/AC.S;

% Gamma parameters
AC.Gamma  = AC.Jx*AC.Jz-AC.Jxz^2;
AC.Gamma1 = (AC.Jxz*(AC.Jx-AC.Jy+AC.Jz))/AC.Gamma;
AC.Gamma2 = (AC.Jz*(AC.Jz-AC.Jy)+AC.Jxz*AC.Jxz)/AC.Gamma;
AC.Gamma3 = AC.Jz/AC.Gamma;
AC.Gamma4 = AC.Jxz/AC.Gamma;
AC.Gamma5 = (AC.Jz-AC.Jx)/AC.Jy;
AC.Gamma6 = AC.Jxz/AC.Jy;
AC.Gamma7 = (AC.Jx*(AC.Jx-AC.Jy)+AC.Jxz*AC.Jxz)/AC.Gamma;
AC.Gamma8 = AC.Jx/AC.Gamma;

% Operating/Trim Conditions
AC.h = 10668;
AC.Va = 267;
AC.alpha = deg2rad(2.6);
%AC.Q = 1382.7035;
%AC.rho = AC.Q*2/(AC.Va)^2;
```

```
45
46 % International Standard Atmosphere
47 AC.rho = 1.225*(1-(2.2558e-5*AC.h))^(4.22559); % Initial Air
      Density (ISA Model)
48
49 % If a dynamic ISA Model is incorporated in simulink we will be
      able to see
50 % dependence on altitude in the results
51
52 AC.Q = 0.5*AC.rho*AC.Va^2; % Initial Dynamic Pressure
53
54 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
55 % Aerodynamic coefficients
56
57 AC.C_L_0          = 0.100;
58 AC.C_L_alpha      = 3.750;
59 AC.C_L_q          = 1.8;
60 AC.C_L_delta_e    = 0.4;
61
62 AC.C_D_0          = 0.0205;
63 AC.C_D_alpha      = 0.300;
64 AC.C_D_q          = 0.0;
65 AC.C_D_delta_e    = -0.10;
66
67 AC.C_m_0          = 0.025;
68 AC.C_m_alpha      = -0.400;
69 AC.C_m_q          = -2.7;
70 AC.C_m_delta_e    = -0.580;
71
72 AC.C_Y_0          = 0.0;
73 AC.C_Y_beta       = -0.68;
74 AC.C_Y_p          = 0.0;
75 AC.C_Y_r          = 0.0;
76 AC.C_Y_delta_a    = 0.016;
77 AC.C_Y_delta_r    = 0.095;
78
79 AC.C_ell_0        = 0.0;
80 AC.C_ell_beta     = -0.080;
81 AC.C_ell_p        = -0.240;
82 AC.C_ell_r        = 0.070;
83 AC.C_ell_delta_a  = -0.042;
84 AC.C_ell_delta_r  = 0.0060;
85
86 AC.C_n_0          = 0.0;
87 AC.C_n_beta       = 0.125;
88 AC.C_n_p          = -0.036;
89 AC.C_n_r          = -0.270;
90 AC.C_n_delta_a    = -0.0010;
```

```
AC.C_n_delta_r   = -0.066;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Trimming the aircraft

% Operating Conditions
AC.pn0     = 0;                          % initial North position
AC.pe0     = 0;                          % initial East position
AC.pd0     = -AC.h;                      % initial Down position (
   negative altitude)
AC.u0      = AC.Va*cos(AC.alpha);        % initial velocity along body
   x-axis
AC.v0      = 0;                          % initial velocity along body
   y-axis
AC.w0      = AC.Va*sin(AC.alpha);        % initial velocity along body
   z-axis
AC.phi0    = 0;                          % initial roll angle
AC.theta0  = AC.alpha;                   % initial pitch angle
AC.psi0    = 0;                          % initial yaw angle
AC.p0      = 0;                          % initial body frame roll rate
AC.q0      = 0;                          % initial body frame pitch
   rate
AC.r0      = 0;                          % initial body frame yaw rate

% Trim control inputs
AC.deltaE = 0.0118078892745844;
AC.deltaT = 20699.2921512690;
AC.deltaA = 0;
AC.deltaR = 0;
```

4 Animation

A simple animation is created in MATLAB for visualization of the aircraft model. This is used for better visualization of the response of the aircraft to various control surface deflections and disturbances. It takes the current state of the aircraft model as the input and updates as the state keeps changing according to the modeled dynamics.

```
function drawAC(uu)
    % process inputs to function
    pn        = uu(1);        % inertial North position
    pe        = uu(2);        % inertial East position
    pd        = uu(3);        % inertial Down position
    u         = uu(4);        % velocity along xb
    v         = uu(5);        % velocity along yb
    w         = uu(6);        % velocity along zb
    phi       = uu(7);        % roll angle
```

```
    theta     = uu(8);        % pitch angle
    psi       = uu(9);        % yaw angle
    p         = uu(10);       % roll rate
    q         = uu(11);       % pitch rate
    r         = uu(12);       % yaw rate
    t         = uu(13);       % time

    % define persistent variables
    persistent uav_handle;
    persistent Vertices
    persistent Faces
    persistent facecolors

    % first time function is called, initialize plot and persistent
        vars
    if t==0
        figure(1), clf
        [Vertices, Faces, facecolors] = defineACBody;
        uav_handle = drawACBody(Vertices,Faces,facecolors,...
                                                pn,pe,pd,phi,theta,
                                                   psi,...
                                                [],'normal');
        title('AC')
        xlabel('East')
        ylabel('North')
        zlabel('-Down')
        view(32,47)  % set the view angle for figure
        axis([-10,10,-10,10,-10,10]);
        grid on
        hold on

    % at every other time step, redraw base and rod
    else
        drawACBody(Vertices,Faces,facecolors,...
                            pn,pe,pd,phi,theta,psi,...
                            uav_handle);
    end
end


%
   ========================================================================

% drawAC
% return handle if 3rd argument is empty, otherwise use 3rd arg as
   handle
%
   ========================================================================
```

```
%
function handle = drawACBody(V,F,patchcolors,...
                                        pn,pe,pd,phi,theta,psi,...
                                        handle,mode)
  V = rotate(V', phi, theta, psi)';  % rotate AC
  V = translate(V', pn, pe, pd)';    % translate AC
  %V = rotate(V', phi, theta, psi)';  % rotate AC


  % transform vertices from NED to XYZ (for matlab rendering)
  % Rotation about x axis by -180 deg then rotation about z' by -90
     deg
  R = [...
      0, 1, 0;...
      1, 0, 0;...
      0, 0, -1;...
      ];
  V = V*R;

  if isempty(handle)
  handle = patch('Vertices', V, 'Faces', F,...
                 'FaceVertexCData',patchcolors,...
                 'FaceColor','flat',...
                 'EraseMode', mode);
  else
    set(handle,'Vertices',V,'Faces',F);
    xlim(handle.Parent, [pe-10,pe+10]);
    ylim(handle.Parent, [pn-10,pn+10]);
    zlim(handle.Parent, [-pd-10,-pd+10]);
    drawnow
  end
end

%%%%%%%%%%%%%%%%%%%%%%%
function XYZ=rotate(XYZ,phi,theta,psi)
  % define rotation matrix
  R_roll = [...
          1, 0, 0;...
          0, cos(phi), -sin(phi);...
          0, sin(phi), cos(phi)];
  R_pitch = [...
          cos(theta), 0, sin(theta);...
          0, 1, 0;...
          -sin(theta), 0, cos(theta)];
  R_yaw = [...
          cos(psi), -sin(psi), 0;...
          sin(psi), cos(psi), 0;...
```

```
          0, 0, 1];
  R = R_roll*R_pitch*R_yaw;
  % rotate vertices
  XYZ = R*XYZ;
end

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% translate vertices by pn, pe, pd
function XYZ = translate(XYZ,pn,pe,pd)
  XYZ = XYZ + repmat([pn;pe;pd],1,size(XYZ,2));
end

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% define AC vertices and faces
function [V,F,colors] = defineACBody()
    % Define the vertices (physical location of vertices)

    fusel1 = 2;
    fusel2 = 0.5;
    fusel3 = 8;
    fusel_w = 2;
    wingspan = 14;
    chord = 2;
    htail_span = 6;
    htail_chord = 1;
    vtail_span = 4;
    vtail_chord = 1.5;
    V = [...
        fusel1      0     0;... % point 1
        fusel2   fusel_w/2   -fusel_w/2;... % point 2
        fusel2  -fusel_w/2   -fusel_w/2;... % point 3
        fusel2  -fusel_w/2    fusel_w/2;... % point 4
        fusel2   fusel_w/2    fusel_w/2;... % point 5
        -fusel3    0                 0;... % point 6
        -0.5,           wingspan/2,   0;...% 7
        -0.5,           -wingspan/2,  0;...% 8
        -(0.5+chord),   -wingspan/2,  0;...% 9
        -(0.5+chord),   wingspan/2,   0;...% 10
       -(fusel3 - htail_chord),  vtail_span/2, 0;...%11
       -(fusel3 - htail_chord), -vtail_span/2, 0;...%12
       -fusel3,                 -vtail_span/2, 0;...%13
       -fusel3,                  vtail_span/2, 0;...%14
       -(fusel3 - vtail_chord), 0,           0;...%15,
       -fusel3,                 0, -vtail_span;...%16
```

```
    ];

    % AC = stlread("pioneer.stl");
    % V = -AC.Points;
    % F = AC.ConnectivityList;
    % colors = [1,1,1];

    % define faces as a list of vertices numbered above
    F = [...
        1, 2, 3, 1;...  % front
        1, 3, 4, 1;...  % back
        1, 4, 5, 1;...
        1, 2, 5, 1;...
        6, 2, 3, 6;...
        6, 3, 4, 6;...
        6, 4, 5, 6;...
        6, 5, 2, 6;...
        7, 8, 9, 10;...% Wing
        11, 12, 13, 14;...% Horizontal Tail
        15, 16, 6, 15;...% Vertical Tail
        ];

    colors = ones(length(F),3);
end
```

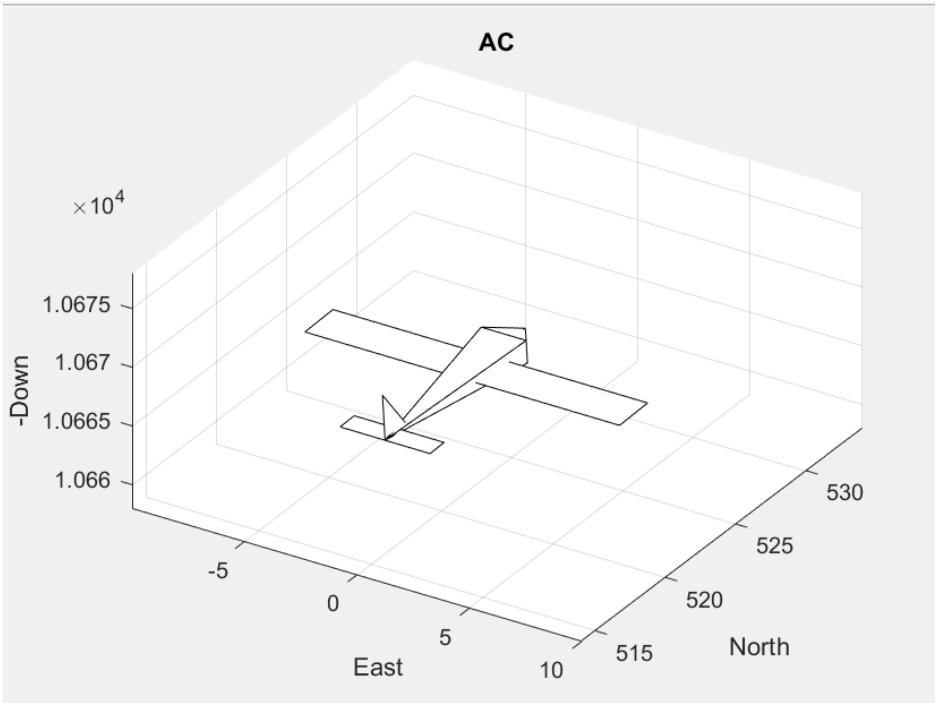


Figure 2: Aircraft Animation

5 Kinematics and Dynamics

5.1 States

A 12-state model of the aircraft was developed using Z-Y-X Brian Trait Euler angle representation, the states being:

1. p_n: x-position in NED inertial frame
2. p_e: y-position in NED inertial frame
3. p_d: z-position in NED inertial frame
4. u: x-velocity in body frame
5. v: y-velocity in body frame
6. w: z-velocity in body frame
7. ϕ: Roll angle
8. θ: Pitch angle
9. ψ: Yaw angle
10. p: Angular rate about x in body frame
11. q: Angular rate about y in body frame
12. r: Angular rate about z in body frame

5.2 System Parameters

1. m: Mass of the aircraft
2. $J = \begin{bmatrix} J_x & J_{xy} & J_{xz} \\ J_{xy} & J_y & J_{yz} \\ J_{xz} & J_{yz} & J_z \end{bmatrix}$ Moment of Inertia Matrix
   Assuming symmetry about $i^b - k^b$ plane: $J_{xy} = Jyz = 0$
3. External forces on the aircraft: $F = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix}$
   External moment: $M = \begin{bmatrix} L \\ M \\ N \end{bmatrix}$
4. $\Gamma = J_x J_z - J_{xz}^2$
   $\Gamma_1 = \frac{J_{xz}(J_x - J_y + J_z)}{\Gamma}$
   $\Gamma_2 = \frac{J_z(J_z - J_y) + J_{xz}^2}{\Gamma}$
   $\Gamma_3 = \frac{J_z}{\Gamma}$
   $\Gamma_4 = \frac{J_{xz}}{\Gamma}$
   $\Gamma_5 = \frac{J_z - J_x}{J_y}$

$\Gamma_6 = \frac{J_{xz}}{J_y}$
$\Gamma_7 = \frac{J_{xz}^2+(J_x-J_y)J_x}{\Gamma}$
$\Gamma_8 = \frac{J_x}{\Gamma}$

5.3 Translational Motion

5.3.1 Kinematics

$$\begin{bmatrix}\dot{p}_n\\ \dot{p}_e\\ \dot{p}_d\end{bmatrix} = \begin{bmatrix}\cos\theta\cos\psi & \sin\phi\sin\theta\cos\phi-\cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi+\sin\phi\sin\psi\\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi+\cos\phi\sin\psi & \cos\phi\sin\theta\sin\psi-\sin\phi\sin\psi\\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta\end{bmatrix}\begin{bmatrix}u\\ v\\ w\end{bmatrix} \tag{1}$$

5.3.2 Dynamics

$$\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{bmatrix} = \begin{bmatrix}rv-qw\\ pw-ru\\ qu-pv\end{bmatrix} + \frac{1}{m}\begin{bmatrix}F_x\\ F_y\\ F_z\end{bmatrix} \tag{2}$$

5.4 Rotational Motion

5.4.1 Kinematics

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}1 & \sin\phi\tan\theta & \cos\phi\tan\theta\\ 0 & \cos\phi & -\sin\phi\\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix} \tag{3}$$

5.4.2 Dynamics

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}\Gamma_1 pq-\Gamma_2 qr\\ \Gamma_5 pr-\Gamma_6(p^2-r^2)\\ \Gamma_7 pq-\Gamma_1 qr\end{bmatrix} + \begin{bmatrix}\Gamma_3 L+\Gamma_4 N\\ \frac{M}{J_y}\\ \Gamma_4 L+\Gamma_8 N\end{bmatrix} \tag{4}$$

Since J_{xz} is not zero there is gyroscopic roll-yaw coupling which can be observed in the simulations. By providing a moment along x-axis there is a yaw rotation and vice versa. Mathematically we can see this effect in $\Gamma_4 L$ and $\Gamma_4 N$ terms in body angular rate derivatives in z and x directions respectively where $\Gamma_4 = \frac{J_{xz}}{\Gamma}$
The Dynamics and Kinematics were modelled using the 6DOF(Euler Angles) Simulink Block available in the Simulink Aerospace Toolbox

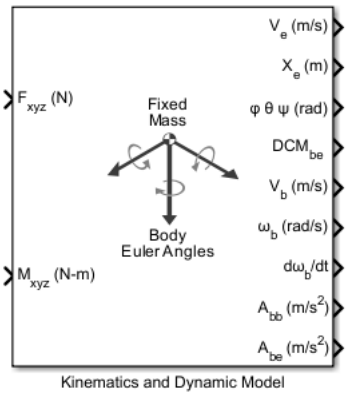


Figure 3: 6DOF (Euler Angles) Simulink Block

6 Forces and Moments

In this section, we compute forces and moments acting on the airframe of the UAV represented in the Body frame. Aerodynamic Forces are computed in the Wind Frame using the experimentally derived parameters of the airframe, Airspeed, and deflections of control surfaces.
The forces acting on an aircraft are gravitational, aerodynamic, and propulsion forces. Moments acting on the aircraft are due to aerodynamics and propulsion.

6.1 Gravitational Model

Gravity in the vehicle frame acts in the +k direction. It is transformed into the body frame of the aircraft. Gravitational forces dont produce moment as our frame is centered at the centre of gravity of the aircraft.

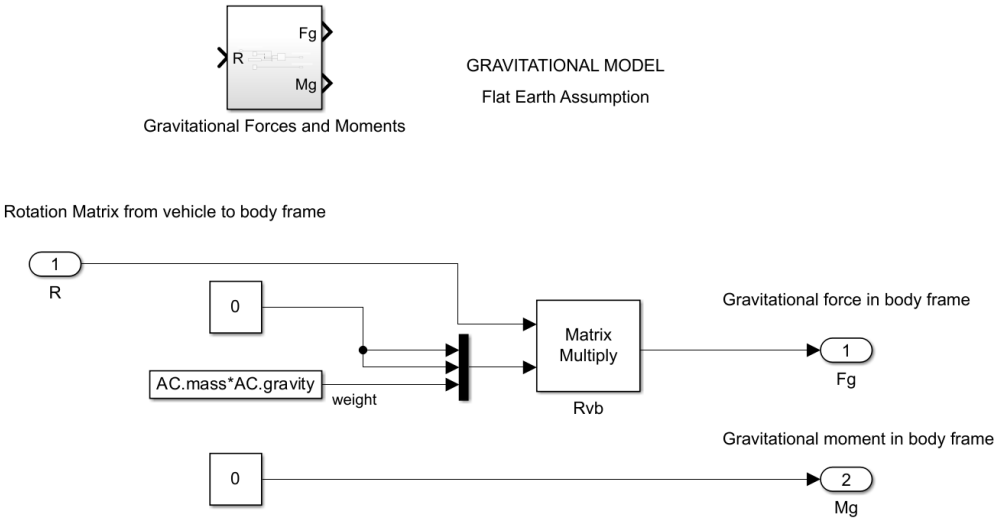


Figure 4: Gravitational Forces and Moments

6.2 Propulsion Model

For the propulsion model, we have assumed that the thrust acts exactly along the i^b direction and there is no moment produced.

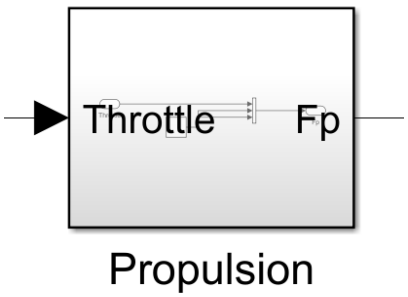


Figure 5: Propulsion Model

6.3 Aerodynamics Model

The Aerodynamics model was built using basic blocks in Simulink using the stability and control derivatives in the aircraft data available for the cruise condition.

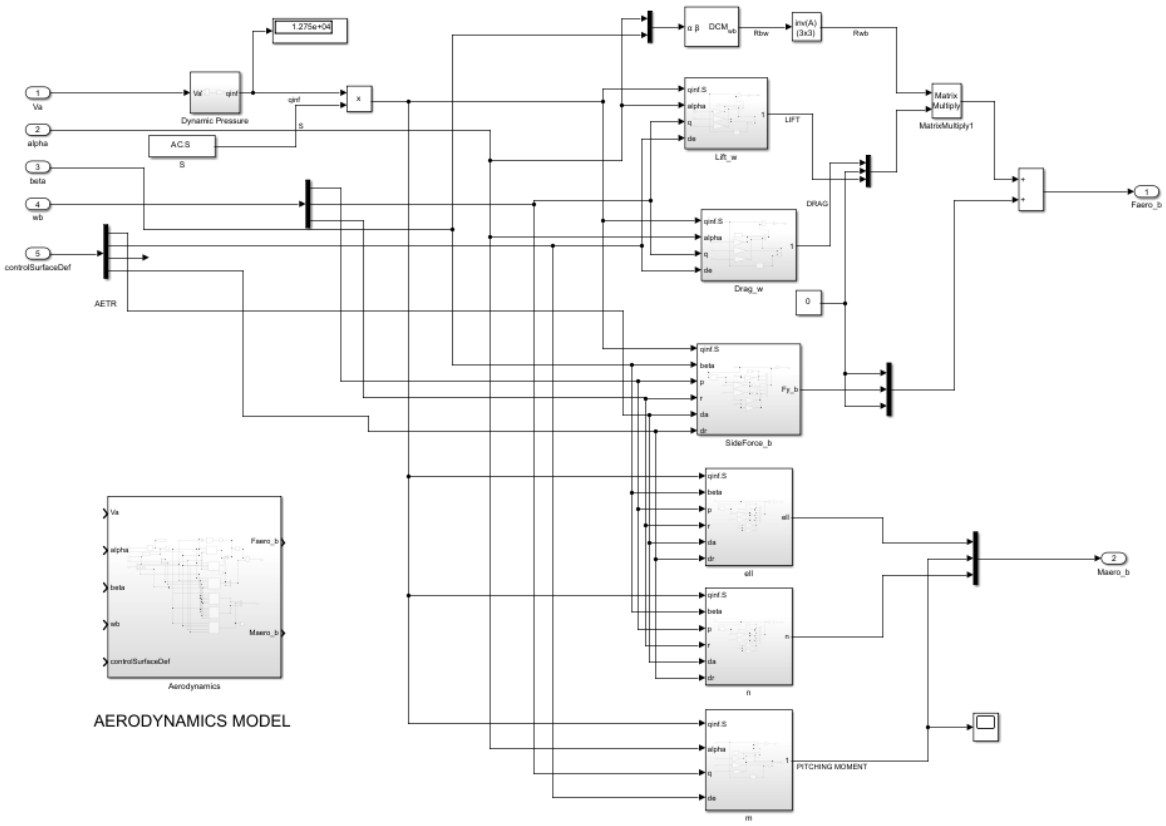


Figure 6: Aerodynamics Model

7 Simulink Model

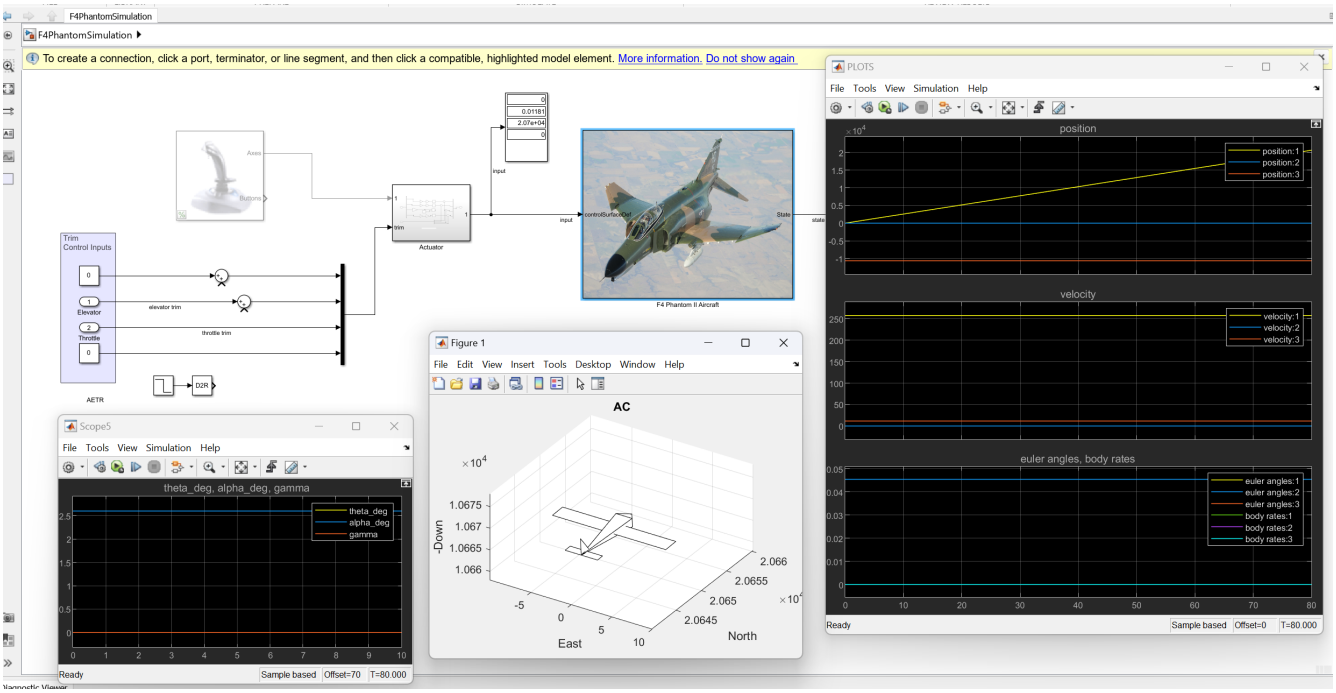


Figure 7: Simulation environment on Simulink

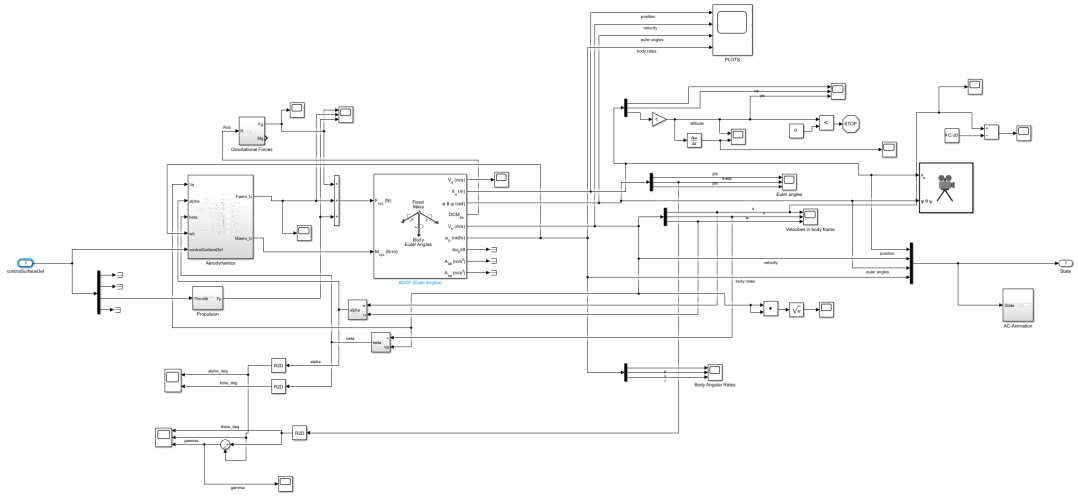


Figure 8: Aircraft Dynamics Block

8 Trimming on Operating Condition

Our operating condition is cruise flight at Velocity of 267 m/s (Mach 0.9) at an altitude of 10688 m. The air density is calculated from the International Standard Atmosphere instead of the one given in the reference.

$$C_{m_o} + C_{m_\alpha}\alpha + C_{m_{\delta e}} = 0 \tag{5}$$

$$\delta_e = -(C_{m_o} + C_{m_\alpha}\alpha)/C_{m\delta e} \tag{6}$$

From caclulation we get $\delta_e = 0.0118$
We trim the non-linear Simulink model on the operating condition using the Model Linearizer app on Simulink.

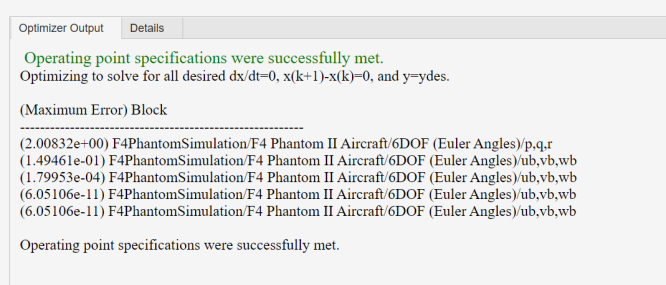
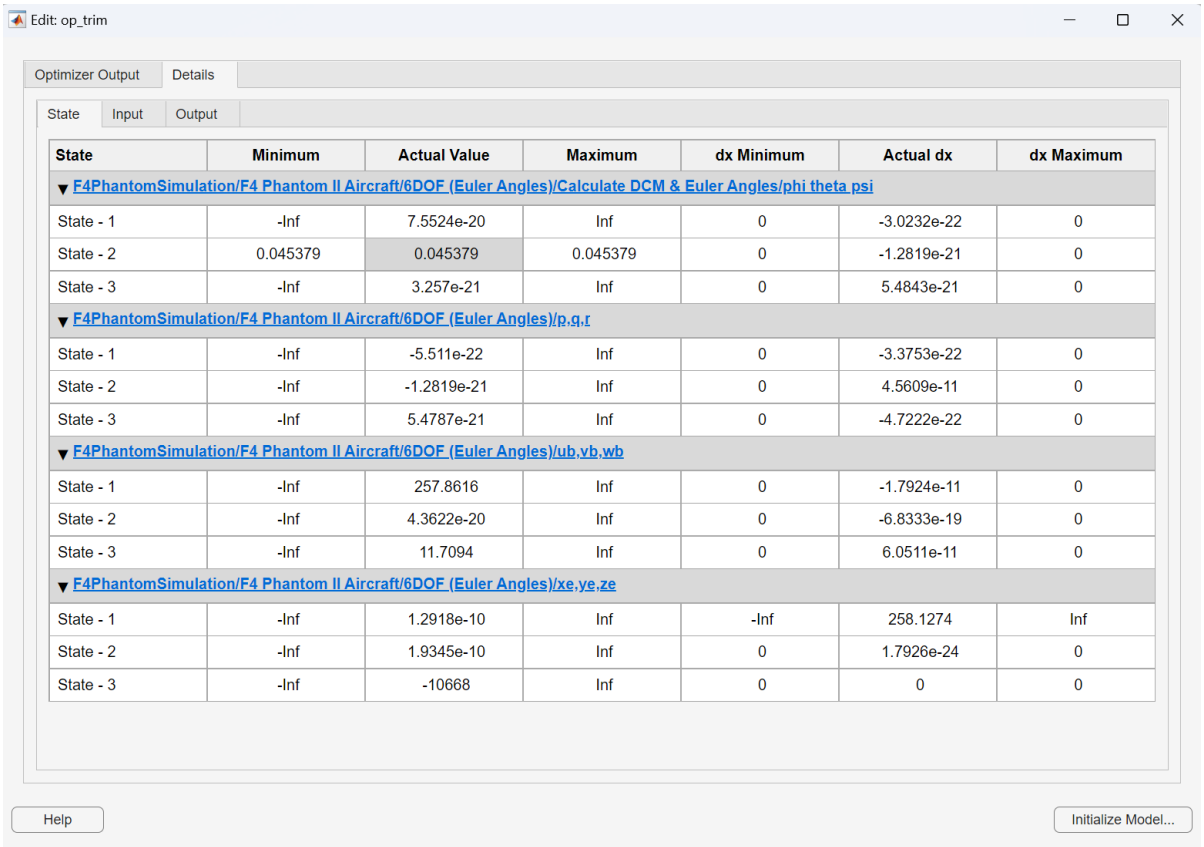


Figure 9: Optimizer results



State	Minimum	Actual Value	Maximum	dx Minimum	Actual dx	dx Maximum
▼ F4PhantomSimulation/F4 Phantom II Aircraft/6DOF (Euler Angles)/Calculate DCM & Euler Angles/phi theta psi						
State - 1	-Inf	7.5524e-20	Inf	0	-3.0232e-22	0
State - 2	0.045379	0.045379	0.045379	0	-1.2819e-21	0
State - 3	-Inf	3.257e-21	Inf	0	5.4843e-21	0
▼ F4PhantomSimulation/F4 Phantom II Aircraft/6DOF (Euler Angles)/p,q,r						
State - 1	-Inf	-5.511e-22	Inf	0	-3.3753e-22	0
State - 2	-Inf	-1.2819e-21	Inf	0	4.5609e-11	0
State - 3	-Inf	5.4787e-21	Inf	0	-4.7222e-22	0
▼ F4PhantomSimulation/F4 Phantom II Aircraft/6DOF (Euler Angles)/ub,vb,wb						
State - 1	-Inf	257.8616	Inf	0	-1.7924e-11	0
State - 2	-Inf	4.3622e-20	Inf	0	-6.8333e-19	0
State - 3	-Inf	11.7094	Inf	0	6.0511e-11	0
▼ F4PhantomSimulation/F4 Phantom II Aircraft/6DOF (Euler Angles)/xe,ye,ze						
State - 1	-Inf	1.2918e-10	Inf	-Inf	258.1274	Inf
State - 2	-Inf	1.9345e-10	Inf	0	1.7926e-24	0
State - 3	-Inf	-10668	Inf	0	0	0

Figure 10: Trimmed states

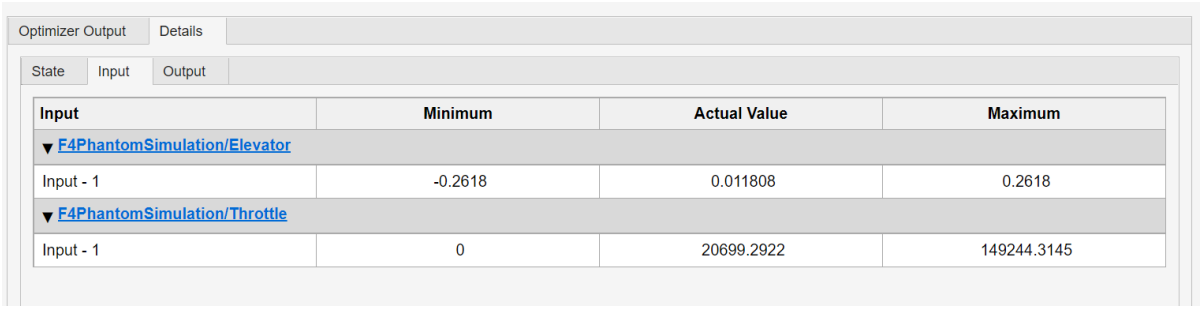


Figure 11: Trimmed Inputs

Note that the trimmed elevator deflection computed numerically by Simulink matches the analytically calculated value.

9 Linearization

Now we linearize the model at this operating condition using the Model Linearizer app. This gives us a linear state space representation of the system.
We reorder the states in the app itself to get the state space representation in the correct order as mentioned in section 5.1.

From the B matrix of the system we can notice that the longitudinal and lateral-directional states are essentially decoupled. The coupling terms are small and can be neglected.

```
B =
                 input(1)   input(2)   input(3)   input(4)
 xe,ye,ze(1)            0          0          0          0
 xe,ye,ze(2)            0          0          0          0
 xe,ye,ze(3)            0          0          0          0
 ub,vb,wb(1)            0       4.19  5.653e-05          0
 ub,vb,wb(2)       0.5679  5.998e-22          0      3.372
 ub,vb,wb(3)            0     -14.02          0          0
 phi theta ps           0          0          0          0
 phi theta ps           0          0          0          0
 phi theta ps           0          0          0          0
 p,q,r (1)         -9.187          0          0       1.54
 p,q,r (2)              0     -10.72          0          0
 p,q,r (3)         0.1055          0          0     -2.603
```

Figure 12: Input matrix of linearised plant

Inputs are ordered as AETR conventional in RC Planes:

1. Aileron deflection
2. Elevator deflection
3. Throttle setting
4. Rudder deflection

The A matrix also shows very small coupling terms in response in states corresponding to perturbations of other states for longitudinal and lateral-directional. So we decouple the dynamics in Longitudinal and Lateral-Directional.

The decoupling is done by subjecting the system dynamics to a similarity transformation with a T matrix. This reorders the states and separates them into Longitudinal and Lateral Directional. Unwanted states are removed from the matrix and a set of 2 state space representations are obtained.

$$z = Tx$$

$$x = T^{-1}z$$

$$\dot{x} = Ax + Bu$$

$$\dot{z} = TAT^{-1}z + TBu$$

$$\dot{z} = \tilde{A}x + \tilde{B}u$$

```
load("linearsys.mat")
plant = linearsys;
A = plant.A;
B = plant.B;

% Similarity Transformation for re-ordering states
T=zeros(12,12);
T(1:3,1:3)=eye(3);
T(4,4) = 1;
T(5,6) = 1;
T(6,11) = 1;
T(7,8) = 1;
T(8,5) = 1;
T(9,10) = 1;
T(10,12) = 1;
T(11,7) = 1;
T(12,9) =1;

Adash = T*plant.A/T;
Adash(4:11,4:11); % Note Longitudinal and Lateral-Directional
   States are almost decoupled
Bdash = T*B;

% Longitudinal Dynamics
Along = Adash(4:7,4:7)
Blong = Bdash(4:7,2:3) % Control input AETR

% Lateral-Directional Dynamics
Alat = Adash(8:11,8:11)
Blat = Bdash(8:11,[1,4])
```

9.1 Longitudinal Dynamics

The Longitudinal states are: u, w, q, θ
Control inputs are δ_e and δ_T

	1	2	3	4
1	-0.0065	0.0199	-11.6821	-9.7999
2	-0.0523	-0.5227	257.2587	-0.4450
3	0.0013	-0.0286	-0.4714	0
4	0	0	1.0000	0

(a) A

	1	2
1	4.1897	0.0001
2	-14.0218	0
3	-10.7194	0
4	0	0

(b) B

Figure 13: Longitudinal State Space Model

Longitudinal dynamics:

$$\begin{bmatrix} \Delta\dot{u} \\ \Delta\dot{w} \\ \Delta\dot{q} \\ \Delta\dot{\theta} \end{bmatrix} = \begin{bmatrix} X_u & X_w & 0 & -g \\ Z_u & Z_w & u_0 & 0 \\ M_u + M_{\dot{w}} Z_u & M_w + M_{\dot{w}} Z_w & M_q + M_{\dot{w}} u_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta w \\ \Delta q \\ \Delta\theta \end{bmatrix} + \begin{bmatrix} X_{\delta e} & X_{\delta T} \\ Z_{\delta e} & Z_{\delta T} \\ M_\delta + M_w \dot{Z}_\delta & M_{\delta T} + M_w \dot{Z}_{\delta T} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\delta e \\ \Delta\delta T \end{bmatrix}$$

9.2 Lateral-Directional Dynamics

The Lateral-Directional states are: v, p, r, ϕ
Control inputs are δ_a and δ_r

	1	2	3	4
1	-0.0980	11.7094	-257.8616	9.7999
2	-0.0695	-1.1971	0.3712	0
3	0.0200	-0.0133	-0.2469	0
4	0	1.0000	0.0454	-0.0000

(a) A

	1	2
1	0.5679	3.3719
2	-9.1866	1.5401
3	0.1055	-2.6032
4	0	0

(b) B

Figure 14: Lateral-Directional State Space Model

Lateral dynamics:

$$\begin{bmatrix} \Delta\dot{v} \\ \Delta\dot{p} \\ \Delta\dot{r} \\ \Delta\dot{\phi} \end{bmatrix} = \begin{bmatrix} Y_v & Y_p & Y_r - u_0 & g\cos\theta_0 \\ L_v & L_p & L_r & 0 \\ N_v & N_p & N_r & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta v \\ \Delta p \\ \Delta r \\ \Delta\phi \end{bmatrix} + \begin{bmatrix} 0 & Y_{\delta r} \\ L_{\delta a} & L_{\delta r} \\ N_{\delta a} & N_{\delta r} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\delta a \\ \Delta\delta r \end{bmatrix}$$

$$\begin{pmatrix} \dot{\tilde{u}} \\ \dot{\tilde{w}} \\ \dot{\tilde{q}} \\ \dot{\tilde{\theta}} \\ \dot{\tilde{h}} \end{pmatrix} = \begin{pmatrix} X_u & X_w & X_q & -g\cos\theta^* & 0 \\ Z_u & Z_w & Z_q & -g\sin\theta^* & 0 \\ M_u & M_w & M_q & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sin\theta^* & -\cos\theta^* & 0 & u^*\cos\theta^* + w^*\sin\theta^* & 0 \end{pmatrix} \begin{pmatrix} \tilde{u} \\ \tilde{w} \\ \tilde{q} \\ \tilde{\theta} \\ \tilde{h} \end{pmatrix} + \begin{pmatrix} X_{\delta_e} & X_{\delta_t} \\ Z_{\delta_e} & 0 \\ M_{\delta_e} & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \tilde{\delta}_e \\ \tilde{\delta}_t \end{pmatrix}, \tag{5.51}$$

$$\begin{pmatrix} \dot{\tilde{v}} \\ \dot{\tilde{p}} \\ \dot{\tilde{r}} \\ \dot{\tilde{\phi}} \\ \dot{\tilde{\psi}} \end{pmatrix} = \begin{pmatrix} Y_v & Y_p & Y_r & A_{14} & 0 \\ L_v & L_p & L_r & 0 & 0 \\ N_v & N_p & N_r & 0 & 0 \\ 0 & 1 & A_{43} & A_{44} & 0 \\ 0 & 0 & A_{53} & A_{54} & 0 \end{pmatrix} \begin{pmatrix} \tilde{v} \\ \tilde{p} \\ \tilde{r} \\ \tilde{\phi} \\ \tilde{\psi} \end{pmatrix} + \begin{pmatrix} Y_{\delta_a} & Y_{\delta_r} \\ L_{\delta_a} & L_{\delta_r} \\ N_{\delta_a} & N_{\delta_r} \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \tilde{\delta}_a \\ \tilde{\delta}_r \end{pmatrix},$$

Table 5.1: Lateral state-space model coefficients

Lateral	Formula
Y_v	$\frac{\rho S b v^*}{4\mathsf{m}V_a^*}\left[C_{Y_p}p^* + C_{Y_r}r^*\right] + \frac{\rho S v^*}{\mathsf{m}}\left[C_{Y_0} + C_{Y_\beta}\beta^* + C_{Y_{\delta_a}}\delta_a^* + C_{Y_{\delta_r}}\delta_r^*\right] + \frac{\rho S C_{Y_\beta}}{2\mathsf{m}}\sqrt{u^{*2}+w^{*2}}$
Y_p	$w^* + \frac{\rho V_a^* S b}{4\mathsf{m}} C_{Y_p}$
Y_r	$-u^* + \frac{\rho V_a^* S b}{4\mathsf{m}} C_{Y_r}$
Y_{δ_a}	$\frac{\rho V_a^{*2} S}{2\mathsf{m}} C_{Y_{\delta_a}}$
Y_{δ_r}	$\frac{\rho V_a^{*2} S}{2\mathsf{m}} C_{Y_{\delta_r}}$
L_v	$\frac{\rho S b^2 v^*}{4V_a^*}\left[C_{p_p}p^* + C_{p_r}r^*\right] + \rho S b v^*\left[C_{p_0} + C_{p_\beta}\beta^* + C_{p_{\delta_a}}\delta_a^* + C_{p_{\delta_r}}\delta_r^*\right] + \frac{\rho S b C_{p_\beta}}{2}\sqrt{u^{*2}+w^{*2}}$
L_p	$\Gamma_1 q^* + \frac{\rho V_a^* S b^2}{4} C_{p_p}$
L_r	$-\Gamma_2 q^* + \frac{\rho V_a^* S b^2}{4} C_{p_r}$
L_{δ_a}	$\frac{\rho V_a^{*2} S b}{2} C_{p_{\delta_a}}$
L_{δ_r}	$\frac{\rho V_a^{*2} S b}{2} C_{p_{\delta_r}}$
N_v	$\frac{\rho S b^2 v^*}{4V_a^*}\left[C_{r_p}p^* + C_{r_r}r^*\right] + \rho S b v^*\left[C_{r_0} + C_{r_\beta}\beta^* + C_{r_{\delta_a}}\delta_a^* + C_{r_{\delta_r}}\delta_r^*\right] + \frac{\rho S b C_{r_\beta}}{2}\sqrt{u^{*2}+w^{*2}}$
N_p	$\Gamma_7 q^* + \frac{\rho V_a^* S b^2}{4} C_{r_p}$
N_r	$-\Gamma_1 q^* + \frac{\rho V_a^* S b^2}{4} C_{r_r}$
N_{δ_a}	$\frac{\rho V_a^{*2} S b}{2} C_{r_{\delta_a}}$
N_{δ_r}	$\frac{\rho V_a^{*2} S b}{2} C_{r_{\delta_r}}$
A_{14}	$g\cos\theta^*\cos\phi^*$
A_{43}	$\cos\phi^*\tan\theta^*$
A_{44}	$q^*\cos\phi^*\tan\theta^* - r^*\sin\phi^*\tan\theta^*$
A_{53}	$\cos\phi^*\sec\theta^*$
A_{54}	$p^*\cos\phi^*\sec\theta^* - r^*\sin\phi^*\sec\theta^*$

Table 5.2: Longitudinal state-space model coefficients

Longitudinal	Formula
X_u	$\frac{u^*\rho S}{\mathsf{m}}\left[C_{X_0} + C_{X_\alpha}\alpha^* + C_{X_{\delta_e}}\delta_e^*\right] - \frac{\rho S w^* C_{X_\alpha}}{2\mathsf{m}} + \frac{\rho S c C_{X_q} u^* q^*}{4\mathsf{m}V_a^*} + \frac{\partial T_p}{\partial u}(\delta_t^*, V_a^*)$
X_w	$-q^* + \frac{w^*\rho S}{\mathsf{m}}\left[C_{X_0} + C_{X_\alpha}\alpha^* + C_{X_{\delta_e}}\delta_e^*\right] + \frac{\rho S c C_{X_q} w^* q^*}{4\mathsf{m}V_a^*} + \frac{\rho S C_{X_\alpha} u^*}{2\mathsf{m}} + \frac{\partial T_p}{\partial w}(\delta_t^*, V_a^*)$
X_q	$-w^* + \frac{\rho V_a^* S C_{X_q} c}{4\mathsf{m}}$
X_{δ_e}	$\frac{\rho V_a^{*2} S C_{X_{\delta_e}}}{2\mathsf{m}}$
X_{δ_t}	$\frac{\partial T_p}{\partial \delta_t}(\delta_t^*, V_a^*)$
Z_u	$q^* + \frac{u^*\rho S}{\mathsf{m}}\left[C_{Z_0} + C_{Z_\alpha}\alpha^* + C_{Z_{\delta_e}}\delta_e^*\right] - \frac{\rho S C_{Z_\alpha} w^*}{2\mathsf{m}} + \frac{u^*\rho S C_{Z_q} c q^*}{4\mathsf{m}V_a^*}$
Z_w	$\frac{w^*\rho S}{\mathsf{m}}\left[C_{Z_0} + C_{Z_\alpha}\alpha^* + C_{Z_{\delta_e}}\delta_e^*\right] + \frac{\rho S C_{Z_\alpha} u^*}{2\mathsf{m}} + \frac{\rho w^* S c C_{Z_q} q^*}{4\mathsf{m}V_a^*}$
Z_q	$u^* + \frac{\rho V_a^* S C_{Z_q} c}{4\mathsf{m}}$
Z_{δ_e}	$\frac{\rho V_a^{*2} S C_{Z_{\delta_e}}}{2\mathsf{m}}$
M_u	$\frac{u^*\rho S c}{J_y}\left[C_{m_0} + C_{m_\alpha}\alpha^* + C_{m_{\delta_e}}\delta_e^*\right] - \frac{\rho S c C_{m_\alpha} w^*}{2J_y} + \frac{\rho S c^2 C_{m_q} q^* u^*}{4J_y V_a^*}$
M_w	$\frac{w^*\rho S c}{J_y}\left[C_{m_0} + C_{m_\alpha}\alpha^* + C_{m_{\delta_e}}\delta_e^*\right] + \frac{\rho S c C_{m_\alpha} u^*}{2J_y} + \frac{\rho S c^2 C_{m_q} q^* w^*}{4J_y V_a^*}$
M_q	$\frac{\rho V_a^* S c^2 C_{m_q}}{4J_y}$
M_{δ_e}	$\frac{\rho V_a^{*2} S c C_{m_{\delta_e}}}{2J_y}$

10 Dynamic Modes and Control Design Models

For designing autopilot for the aircraft we need reduced order model of the full nonlinear 12 state model. We linearise the model at trim condition and reduce it to get a set of 2 state space models for longitudinal and lateral directional dynamics. We then find out transfer functions relating the control inputs to the states and create cascaded closed loops by successive loop closure.
The basic idea behind successive loop closure is to close several simple feedback loops in succession around the open-loop plant dynamics (1st or 2nd order) rather then designing a single (presumably more complicated) control systems.

10.1 Longitudinal Dynamics

Using the Longitudinal Dynamics system the eigen values of the A matrix can be computed which will give us the poles for the Short Period mode and the Phugoid mode

```
load("F4Parameters.mat")
load('F4LongitudinalDynamics.mat')
C = eye(4); D=0;
longsys = ss(Along,Blong,C,D);

figure;
title("Longitudinal Dynamics pole-zero plot")
pzplot(longsys,'b')

Glong = tf(longsys);

% Elevator Input
Ge2u = Glong(1,1);
Ge2w = Glong(2,1);
Ge2q = Glong(3,1);
Ge2theta = Glong(4,1);

% Throttle Input
Gt2u = Glong(1,2);
Gt2w = Glong(2,2);
Gt2q = Glong(3,2);
Gt2theta = Glong(4,2);
```

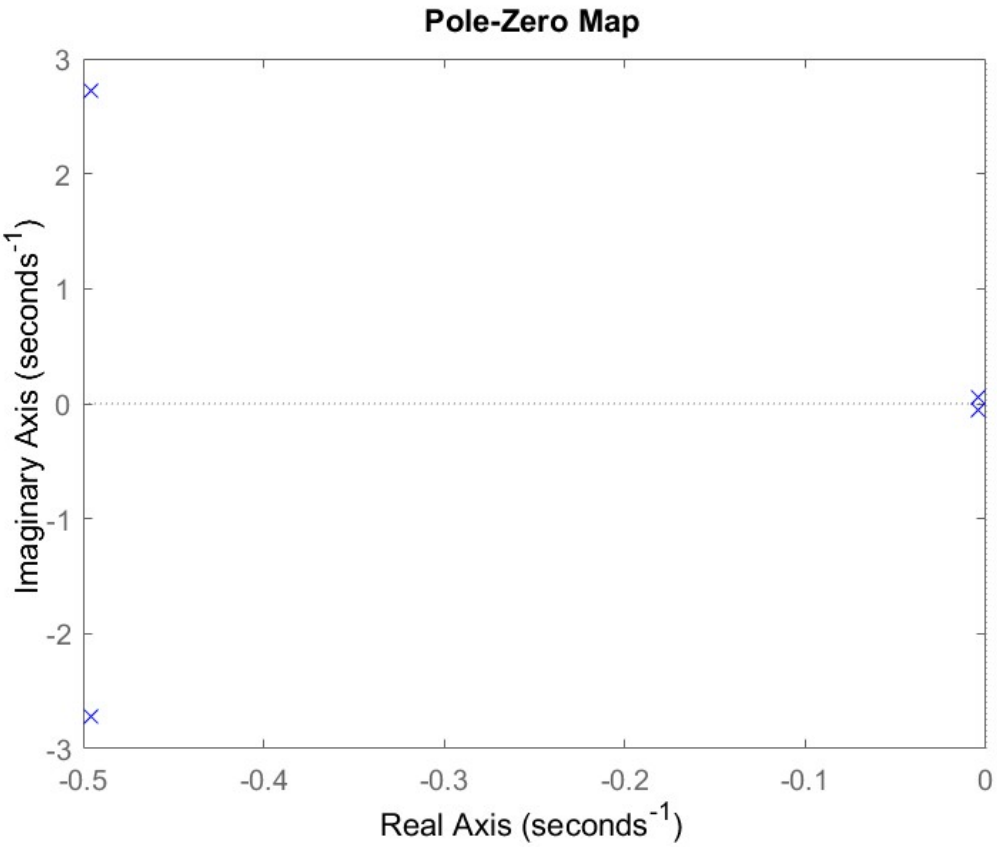


Figure 17: Longitudinal Dynamics Pole-Zero Plot

There are 2 sets of oscillatory poles as seen in the pole-zero map, All 4 are stable LHP poles. The set closer to origin(small ω_n) and with small $\zeta\omega_n$ which shows that the time period of oscillations will be large and the decay will also be slow (envelope of 2nd order response depends on $\zeta\omega_n$ term). This is the Phugoid Mode.

Phugoid Mode
-0.0043 + 0.0526j
0.0043 - 0.0526j

Time period = 119.45s (around 2 minutes)

The other set of poles further from the orgin and the real axis belong to the Short Period mode which have a fast transient response and decay faster on perturbation.

Short Period Mode
-0.4960 + 2.7151j
-0.4960 - 2.7151j

Time period = 2.31s

10.2 Lateral-Directional Dynamics

Using the Lateral-Directional Dynamics system the eigen values of the A matrix can be computed which will give us the poles for the lateral-directional modes: Spiral Mode, Roll Subsidence Mode, Dutch Roll Mode

```
load("F4Parameters.mat")
load('F4LateralDynamics.mat')
C = eye(4); D=0;
latsys = ss(Alat,Blat,C,D);

figure;
pzplot(latsys,'r')

Glat = tf(latsys);

% Aileron Input
Ga2v = Glat(1,1);
Ga2p = Glat(2,1);
Ga2r = Glat(3,1);
Ga2phi = Glat(4,1);

% Rudder Input
Gr2v = Glat(1,2);
Gr2p = Glat(2,2);
Gr2r = Glat(3,2);
Gr2phi = Glat(4,2);
```

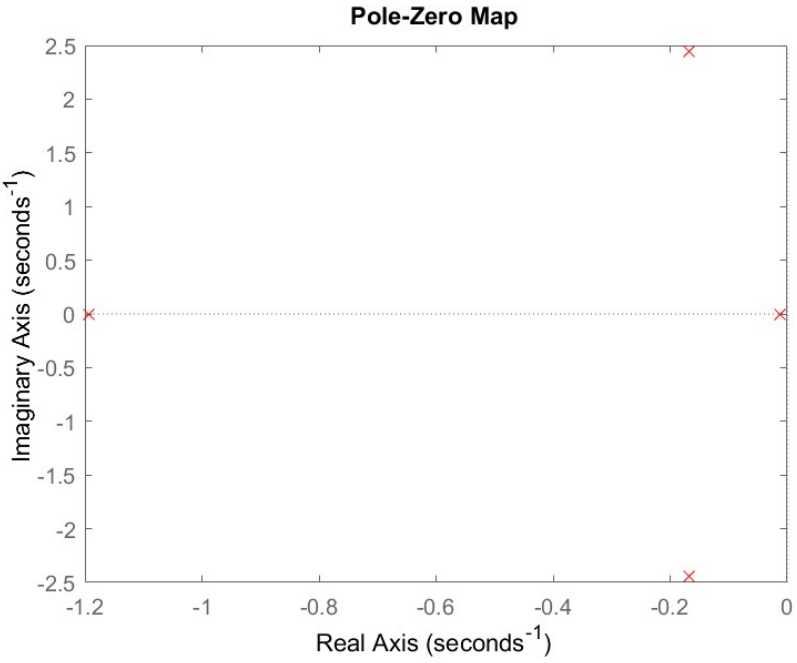


Figure 18: Lateral-Directional Dynamics Pole-Zero Plot

There are 2 non-oscillatory poles. The one further away from origin is a very stable fast pole corresponding to the roll subsidence mode. The one closer to the origin is a stable but slow pole corresponding to the Spiral mode. The oscillatory poles make up the Dutch Roll mode.

Roll Subsidence Mode
-1.1934
Spiral Mode
-0.0118
Dutch Roll Mode
-0.1684 + 2.4431j
-0.1684 - 2.4431j

11 Influence of Stability Derivatives on Eigen Values

11.1 Longitudinal

The longitudinal eigenvalues are plotted for different values of C_{m_q} in Figure 19. It can be observed that increasing the magnitude of C_{m_q} stabilizes the dynamics of the short-period mode while having no observable effect on the phugoid dynamics.

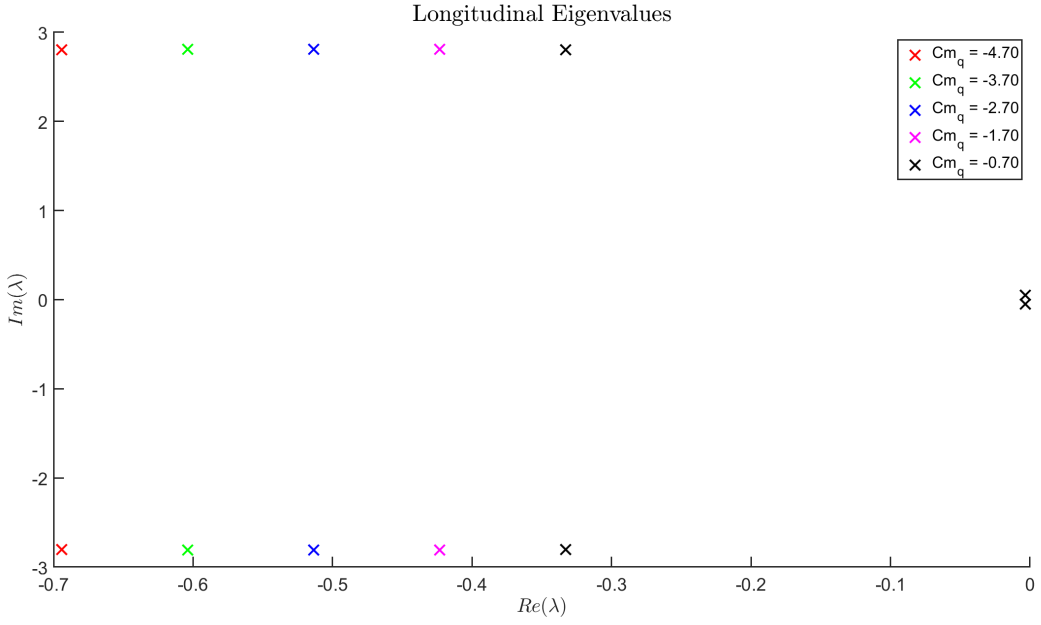


Figure 19: Effect of C_{m_q} on longitudinal dynamics

Figure 20 show the longitudinal eigenvalues for different values of C_{x_u}. At first glance, there seems to be no observable effect on any of the modes. However, a closer look in Figure 21 reveals that an increase in the magnitude of C_{x_u} stabilizes the phugoid mode dynamics.

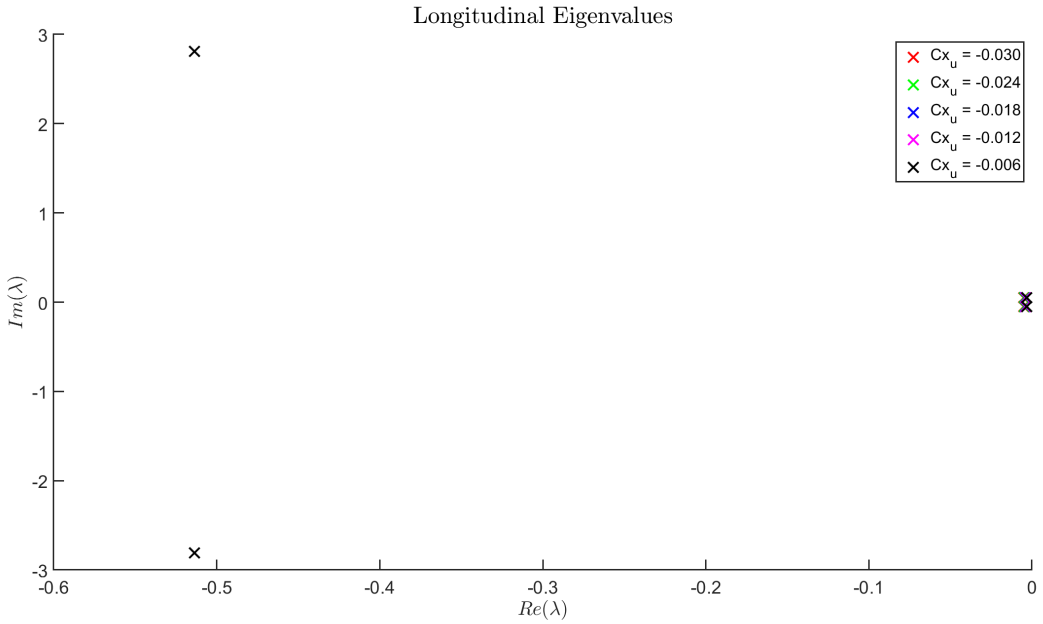


Figure 20: Effect of C_{x_u} on longitudinal dynamics

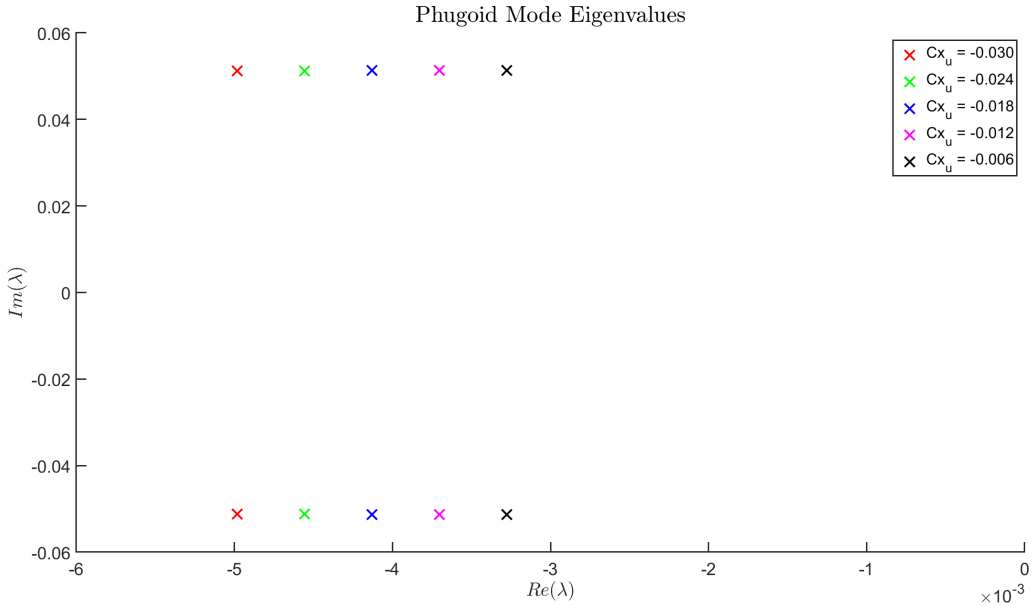


Figure 21: Effect of C_{x_u} on phugoid mode dynamics

11.2 Lateral - Directional

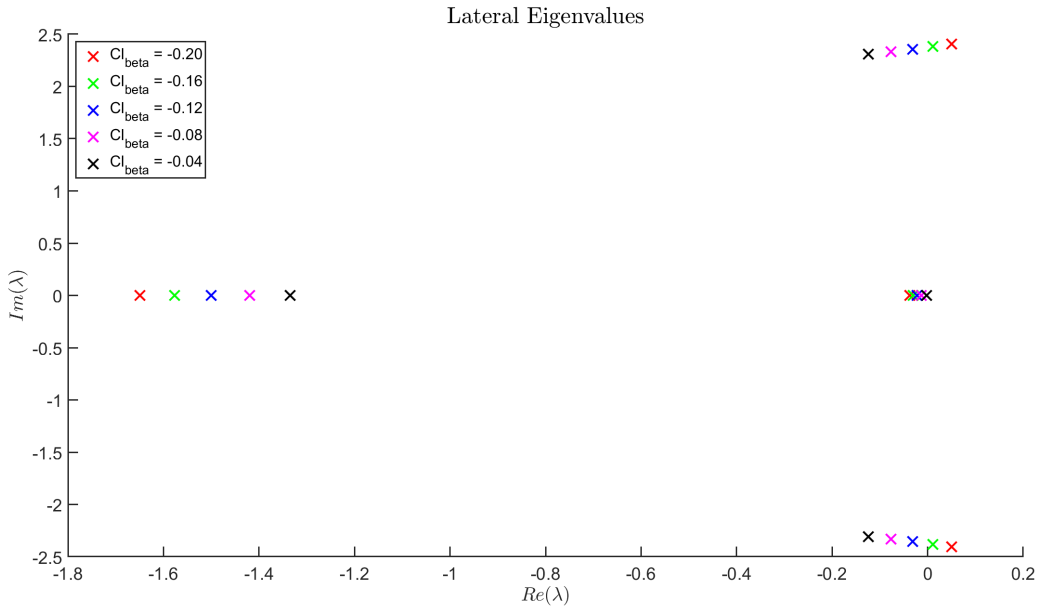


Figure 22: Effect of C_{l_β} on lateral-directional dynamics

Figures 22 and 23 show the lateral-directional eigenvalues plotted for different values of C_{l_β} and C_{n_r} respectively. It can be observed that an increase in the magnitude of C_{l_β} stabilizes the roll and spiral modes, but the dutch-roll mode becomes unstable. This is an effect of the roll stability significantly outweighing the yaw stability. On the other hand, increasing the magnitude of C_{n_r} the dutch-roll and spiral modes while having insignificant effect on the roll dynamics.

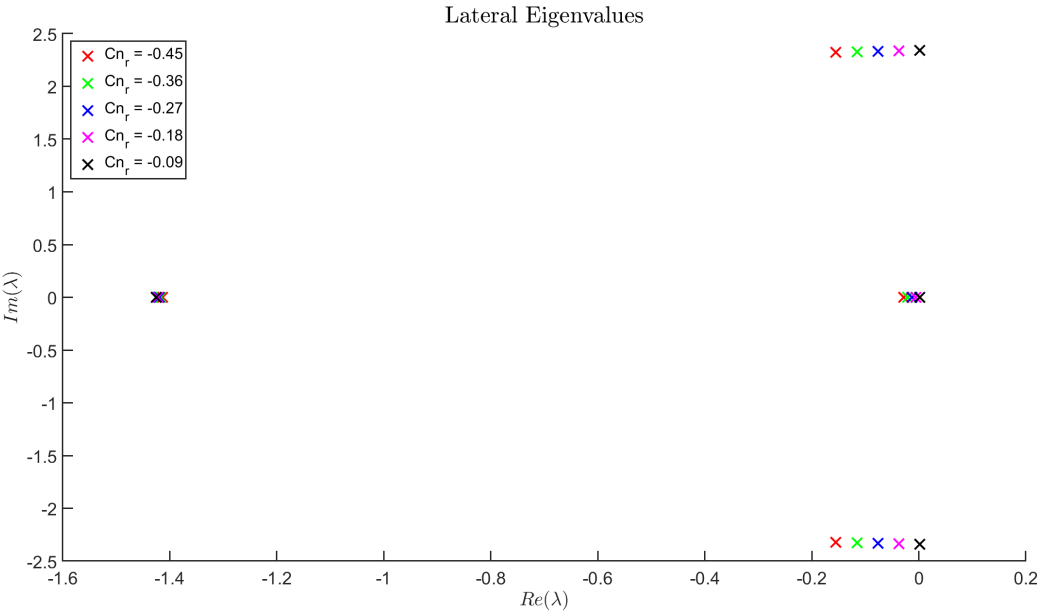


Figure 23: Effect of C_{n_r} on lateral-directional dynamics

12 Transfer Functions

Note that the control inputs here are perturbations on the trim control inputs and the responses are also perturbations on the trim states.

12.0.1 Elevator Deflection Input

$$\text{Ge2u} = \frac{4.19s^3 + 129.1s^2 + 142.8s + 51.02}{s^4 + 1.001s^3 + 7.629s^2 + 0.0689s + 0.02124}$$

$$\text{Ge2w} = \frac{-14.02s^3 - 2765s^2 - 18.73s - 5.641}{s^4 + 1.001s^3 + 7.629s^2 + 0.0689s + 0.02124}$$

$$\text{Ge2q} = \frac{-10.72s^3 - 5.266s^2 - 0.03633s - 3.468 \times 10^{-18}}{s^4 + 1.001s^3 + 7.629s^2 + 0.0689s + 0.02124}$$

$$\text{Ge2}\theta = \frac{-10.72s^2 - 5.266s - 0.03633}{s^4 + 1.001s^3 + 7.629s^2 + 0.0689s + 0.02124}$$

12.0.2 Throttle setting input

$$\text{Gt2u} = \frac{5.653 \times 10^{-5}s^3 + 5.619 \times 10^{-5}s^2 + 0.00043s - 7.197 \times 10^{-7}}{s^4 + 1.001s^3 + 7.629s^2 + 0.0689s + 0.02124}$$

$$\text{Gt2w} = \frac{-2.955 \times 10^{-6}s^2 + 1.75 \times 10^{-5}s - 3.268 \times 10^{-8}}{s^4 + 1.001s^3 + 7.629s^2 + 0.0689s + 0.02124}$$

$$\text{Gt2q} = \frac{7.344 \times 10^{-8}s^2 + 1.229 \times 10^{-7}s + 5.544 \times 10^{-27}}{s^4 + 1.001s^3 + 7.629s^2 + 0.0689s + 0.02124}$$

$$\text{Gt2}\theta = \frac{7.344 \times 10^{-8}s + 1.229 \times 10^{-7}}{s^4 + 1.001s^3 + 7.629s^2 + 0.0689s + 0.02124}$$

12.0.3 Aileron Deflection Input

$$\text{Ga2v} = \frac{0.5679s^3 - 134s^2 - 180s - 21.73}{s^4 + 1.542s^3 + 6.417s^2 + 7.232s + 0.08423}$$

$$\text{Ga2p} = \frac{-9.187s^3 - 3.169s^2 - 45.75s + 0.07856}{s^4 + 1.542s^3 + 6.417s^2 + 7.232s + 0.08423}$$

$$\text{Ga2r} = \frac{0.1055s^3 + 0.2702s^2 - 2.029s - 1.73}{s^4 + 1.542s^3 + 6.417s^2 + 7.232s + 0.08423}$$

$$\text{Ga2phi} = \frac{-9.182s^2 - 3.157s - 45.84}{s^4 + 1.542s^3 + 6.417s^2 + 7.232s + 0.08423}$$

12.0.4 Rudder Deflection Input

$$\text{Gr2v} = \frac{3.372s^3 + 694.2s^2 + 816.9s - 7.139}{s^4 + 1.542s^3 + 6.417s^2 + 7.232s + 0.08423}$$

$$\text{Gr2p} = \frac{1.54s^3 - 0.6694s^2 - 38.78s + 0.06677}{s^4 + 1.542s^3 + 6.417s^2 + 7.232s + 0.08423}$$

$$\text{Gr2r} = \frac{-2.603s^3 - 3.324s^2 - 1.981s - 1.47}{s^4 + 1.542s^3 + 6.417s^2 + 7.232s + 0.08423}$$

$$\text{Gr2phi} = \frac{1.422s^2 - 0.8203s - 38.87}{s^4 + 1.542s^3 + 6.417s^2 + 7.232s + 0.08423}$$

13 Response to Elevator Deflection

The following plots show repsonse of aircraft Simulink model to 1 deg step deflection in elevator from the trim position. The step deflection takes place at 10s and the simulation is run for 265.4s

13.1 Δu Response

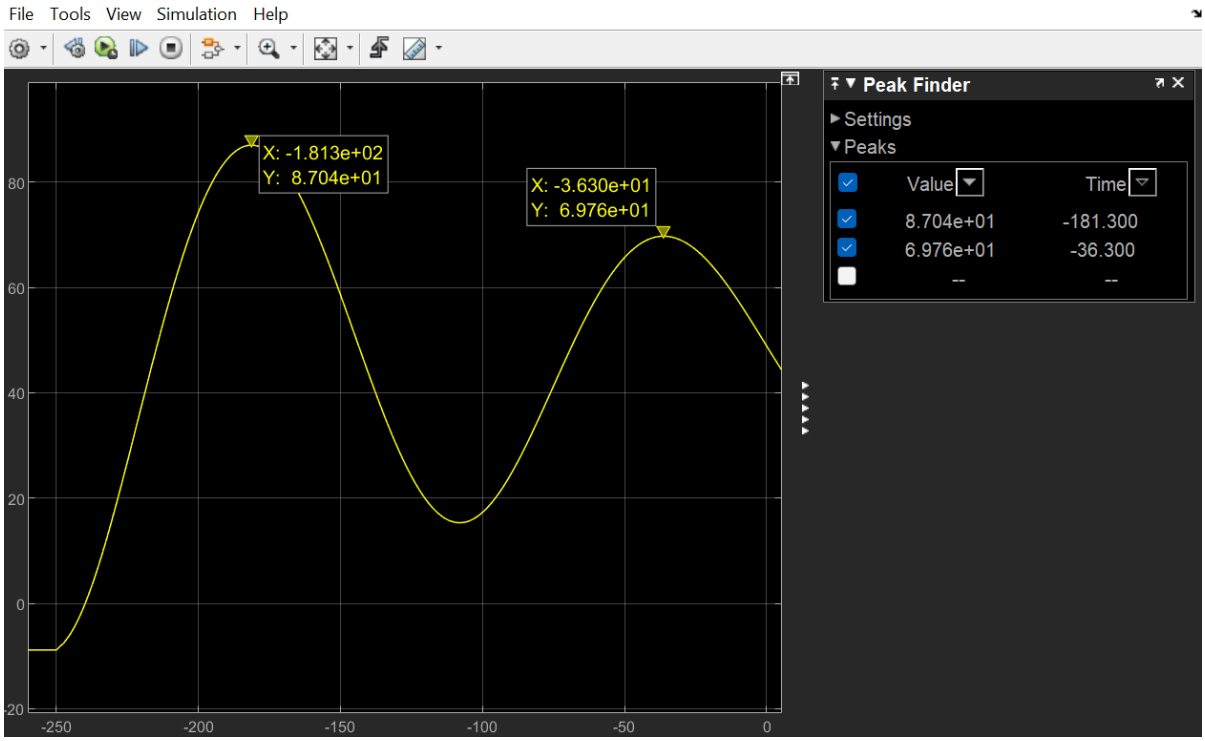


Figure 24: Δu(t)

13.2 $\Delta\gamma$ Response

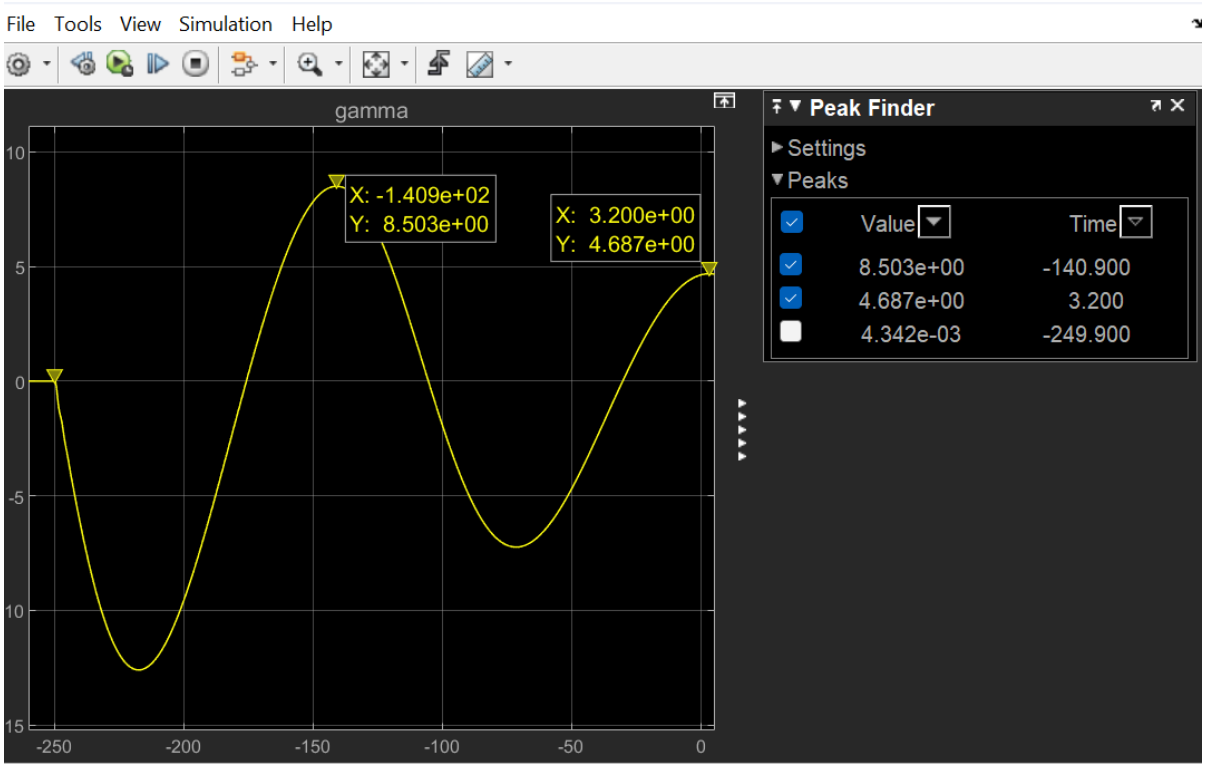


Figure 25: $\Delta\gamma$(t)

Note there is no significant short term response. Phugoid mode is dominant in delta u and delta gamma response.

13.3 α Response

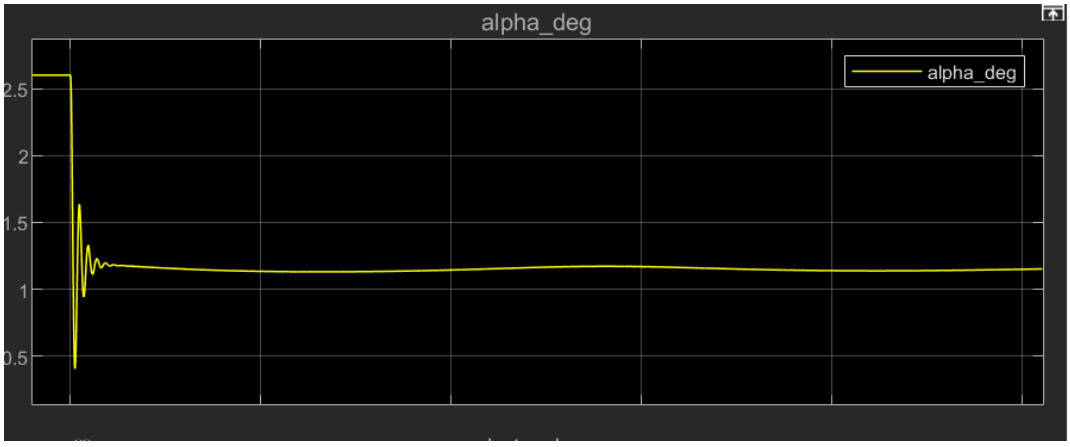


Figure 26: α(t) Long Term

13.4 $\Delta\theta$ Response

13.4.1 Long Term Response

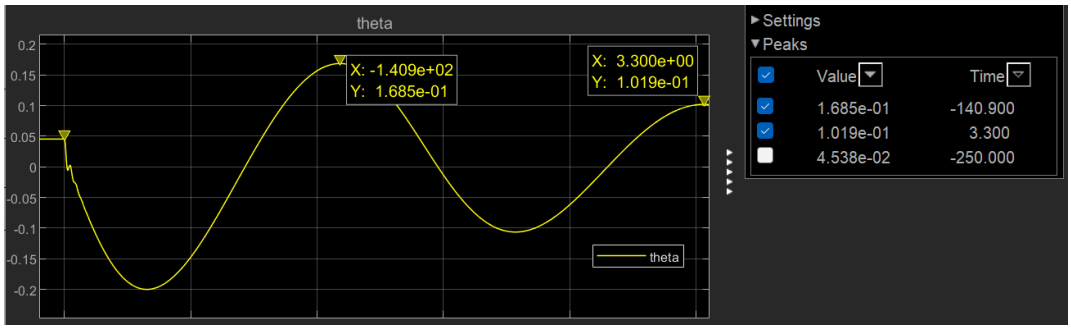


Figure 27: $\Delta\theta$(t) Long Term

13.4.2 Short Term Response

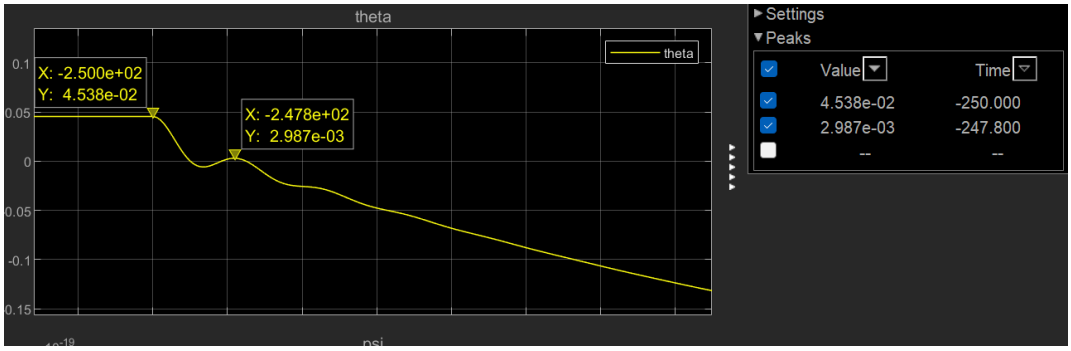


Figure 28: $\Delta\theta$(t) Short Term

Note that long term response shows effect of phugoid while the short term response shows that there is fast decaying oscillations which corresponds to the short period mode. Similar response can be seen in the delta q response

13.5 Δq Response

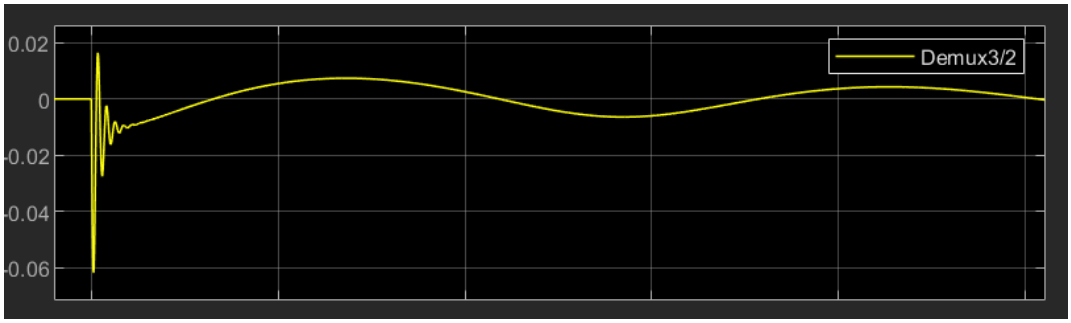


Figure 29: Δq(t)

13.6 MATLAB Transfer Function Responses

Ploting responses based on the transfer functions we have derived for the linearised model at the trim operating condition.
Here the deflection starts at 100s and the simulation stop time is 1000s

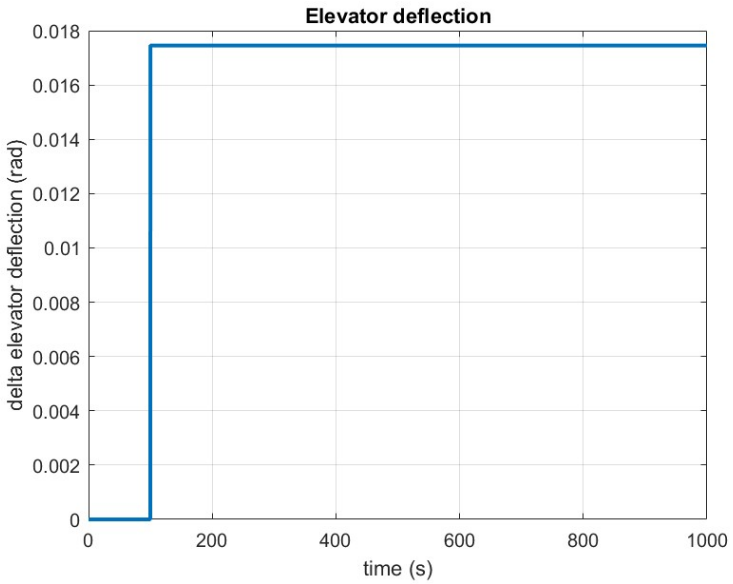
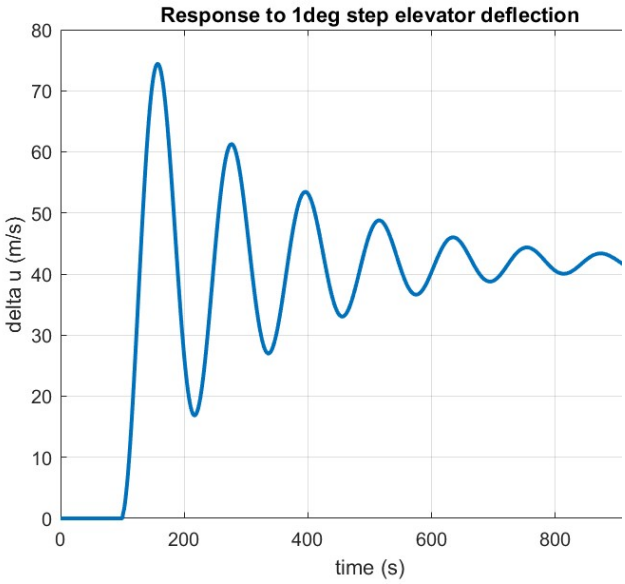
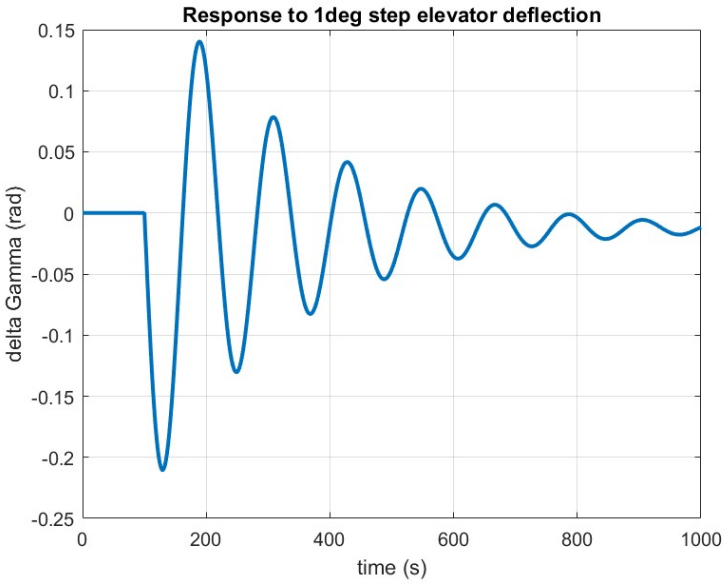


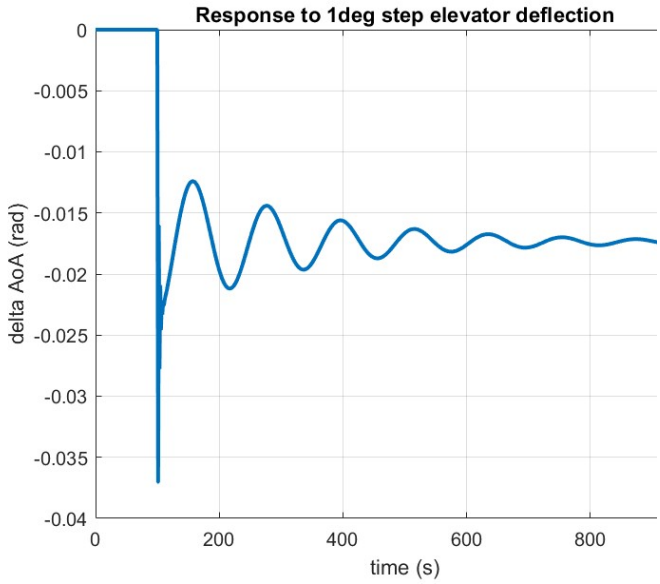
Figure 30: $\Delta\delta_e$(t)



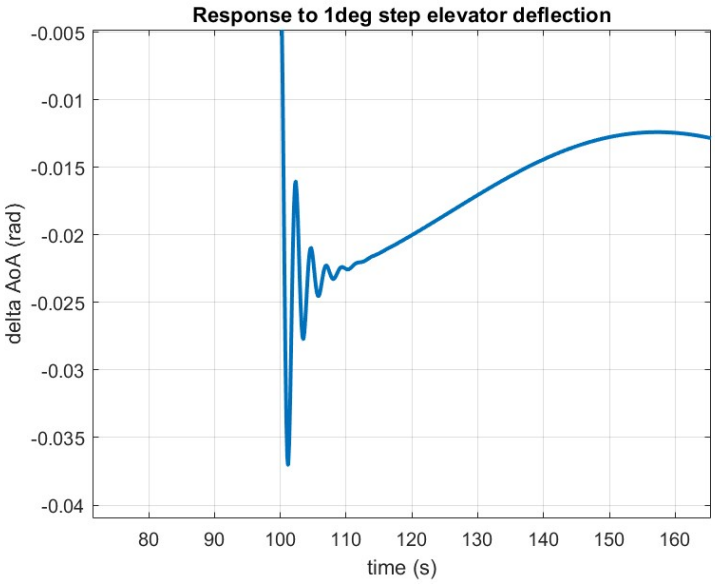
(a) Δu(t)



(b) $\Delta\gamma$(t)



(a) $\Delta\alpha$(t) Long term



(b) $\Delta\alpha$(t) Short term

We can observe that there are two different transients one is a large amplite slow oscillationwhich corresponds to the phugoid mode and then there is a faster much smaller transient due to the Short Period mode seen in delta alpha.

These plots were obtained using the lsim function of MATLAB with transfer functions obtained earlier and a custom mad step input. This shows the expected open loop response of plant when we design our Autopilot based on the control model.

14 Response to Aileron Deflection

a $-15°$ step response was subjected to the plant and the response of (u,v,w) velocity in body frame, (p,q,r) body angular rates and (ϕ, θ, ψ) were plotted.

In the Simulink model I added a step response of $-15°$ in aileron deflection. The aircraft crashes into the ground at 58.156s.

14.1 Velocity Response

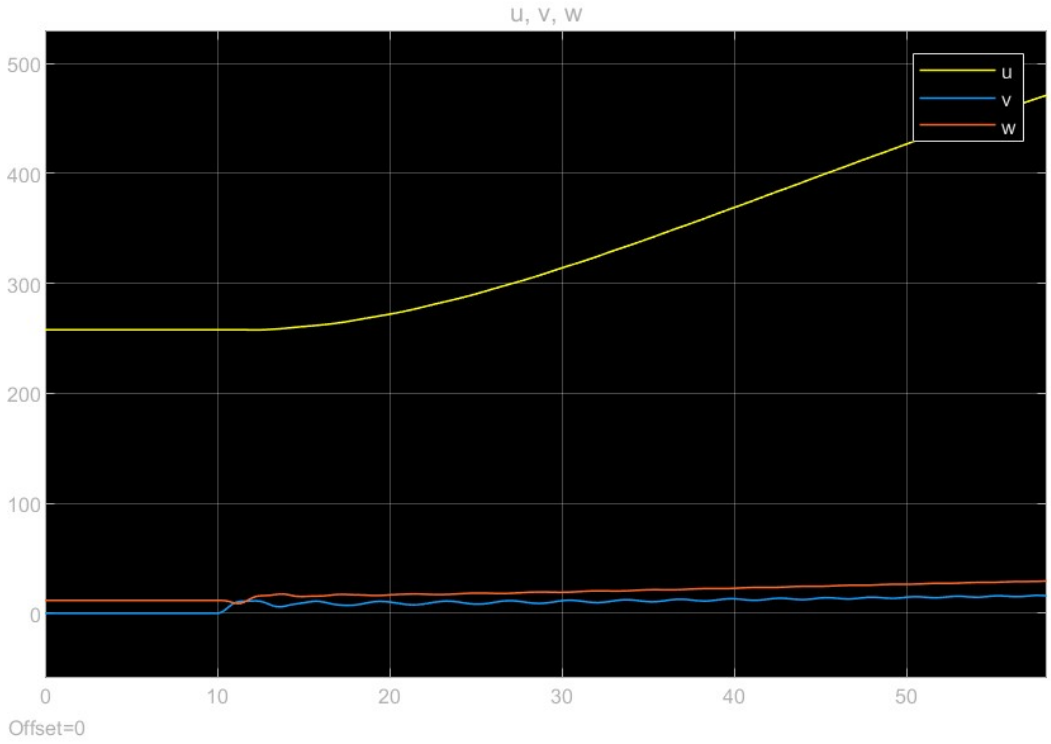
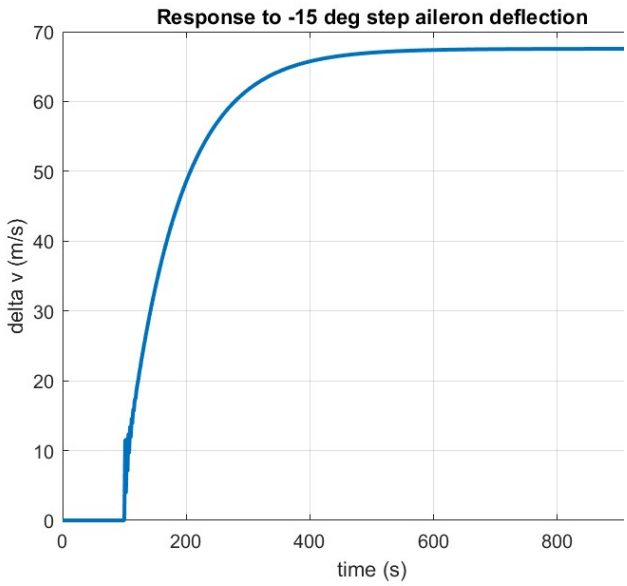


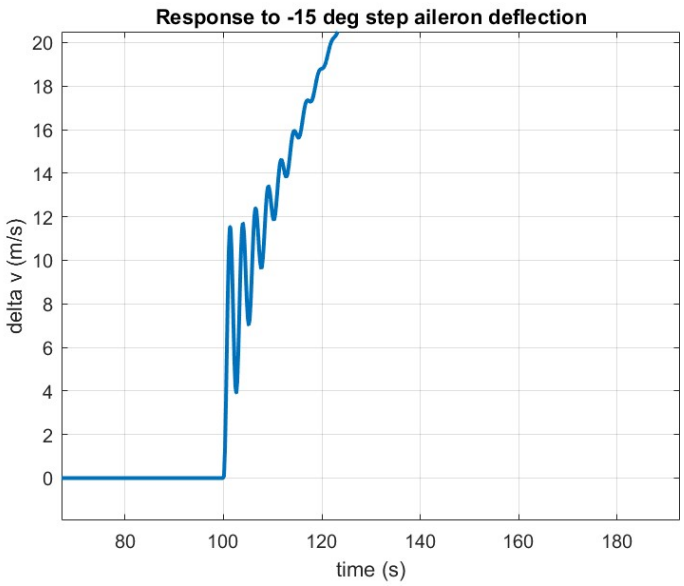
Figure 33: Velocity components in body frame

delta v(t) expected from transfer function relating aileron deflection to delta v looks like a first order response so we believe that it is being dominated by a slow first order system which is the Spiral Mode

The delta v response shows comparatively faster transient oscillations at start which corresponds to Dutch Roll mode. Note that an aileron deflection will excite all the lateral longitudinal modes, there will not be any pure modal response.



(a) Δv(t)



(b) Δv(t) Short Term

14.2 Body Angular Rates Response

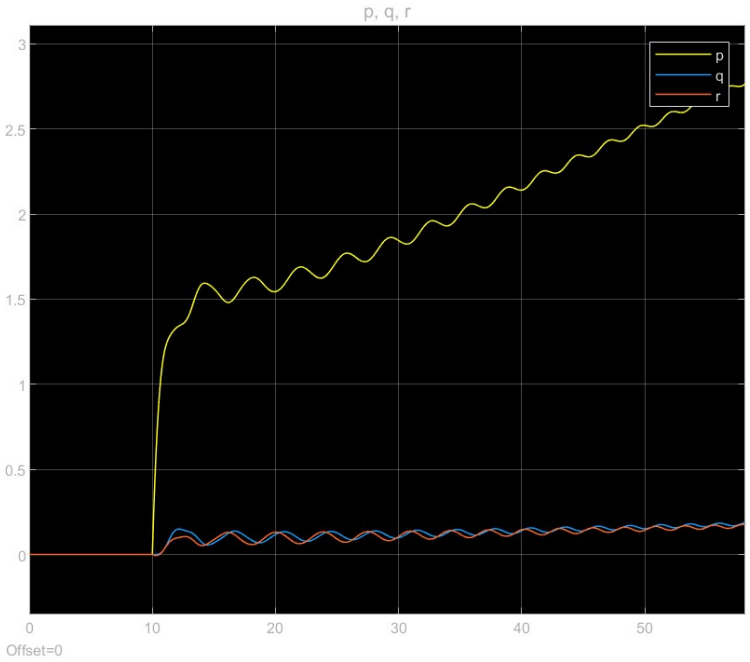
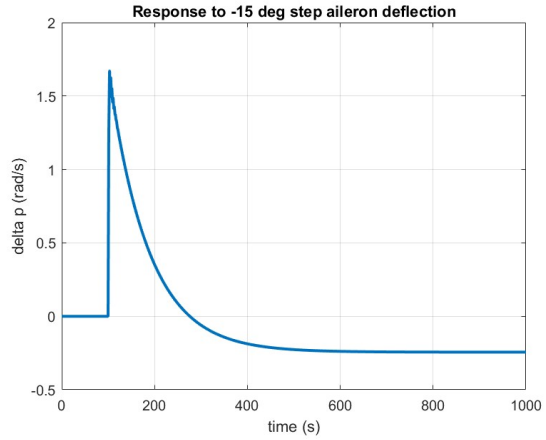
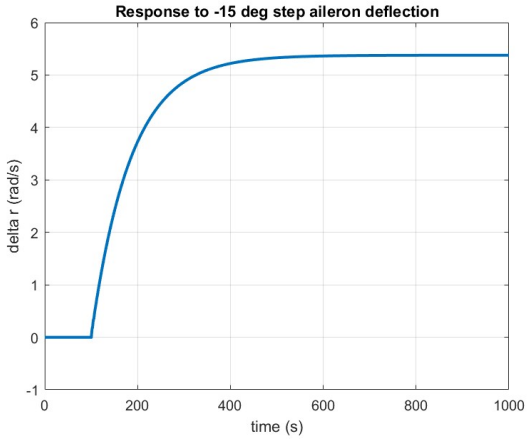


Figure 35: Body angular rate components in body frame



(a) Δp(t)



(b) Δr(t)

14.3 Attitude angles Response

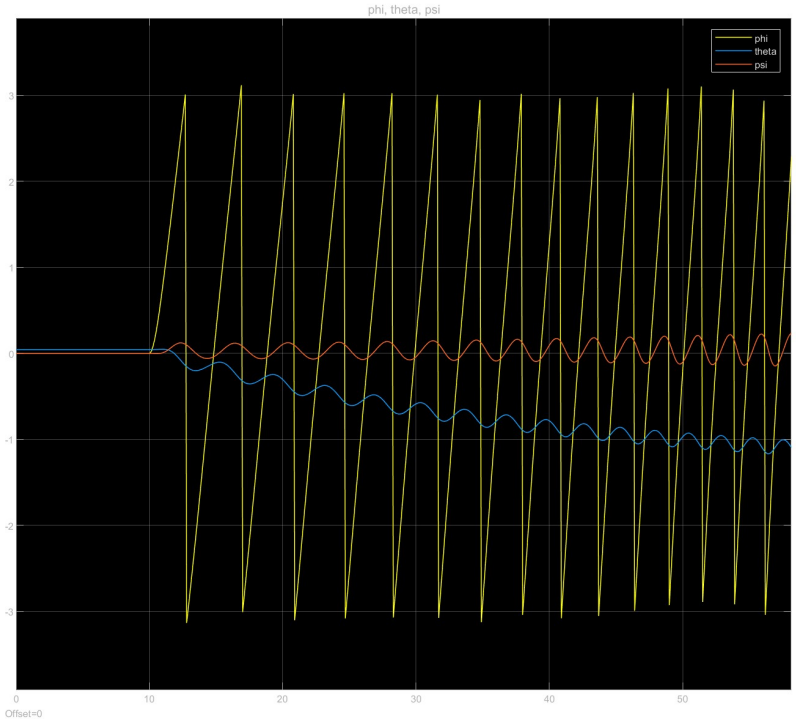


Figure 37: attitude angles

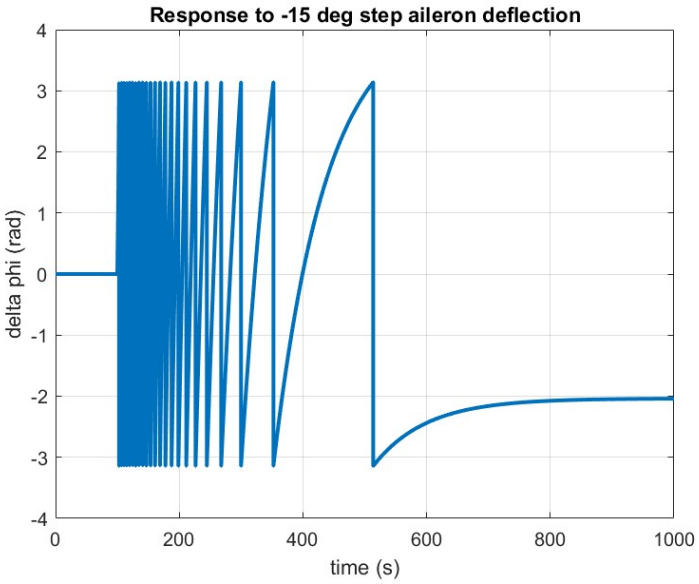


Figure 38: $\Delta\phi$(t)

This seems to have oscillations because phi is wrapped between 0 to pi

15 Conclusion

The full 12 state linearised state space space representation was decoupled into longitudinal and lateral-directional dynamics. The eigen values of the system A matrix gave us the poles of the system which corresponds to the dynamic modes of the aircraft. Longitudinal system contained a set of 2 oscillatory poles for Phugoid and Short-Period Mode and the Lateral-Directional Dynamics had 2 real poles corresponding to roll subsidence and spiral mode and an oscialltory pole representing Dutch Roll mode.
From the eigen values we were able to observe that the aircraft without an autpilot is stable for small pertubations about the trimmed condition. but the poles for phugoid and spiral are indeed very close to the orgin. Varying the stability derivatives we could see the shift in poles and relate it back to theoretical understanding of which stability derivative affects which mode. Step deflections to elevator and ailerons was given in the model and we observed the effect it has on the flight.
An actuator block was implemented and using Simulink's Joystick input block we could actually pilot the aircraft in the simulation model. The future plan is to use the reduced order control model to design autopilot loops and them implement contorlleers with the same gain in the simulation model and observe the effects and handling quality while flying.

References

[1] Airplane Flight Dynamics and Automatic Flight Controls, Jan Roskam

[2] Small Unmanned Aircraft, Randall Tim
https://github.com/randybeard/mavsim_public

[3] Aircraft Flight Mechanics Notes, Harry Smith
https://www.aircraftflightmechanics.com/NotesIntroduction.html

[4] Coupling Dynamics in Aircraft: A Historical Perspective, Richard E.Day
https://ntrs.nasa.gov/api/citations/19970019603/downloads/19970019603.pdf