

---

## Table of Contents

Assignment 1 .....	1
Problem 3 .....	1
Problem 4 .....	2
Problem 5 .....	3
Problem 6 .....	4
Problem 7 .....	4
Problems .....	5
Functions .....	10

## Assignment 1

Aero 707 Liam Hood

```
clear; close all; clc
global r2d
r2d = 180/pi;
% %% Problem 2
% problem_2()
%
```

## Problem 3

```
global r2d
fprintf("\n*****Problem 3*****\n")
% 2.3-1 Find instantaneous angle of attack and angle of side slip
% in the gusts

% Given starting conditions
v = 500;
alpha = 8/r2d;
beta = -5/r2d;
% Rotation matrices
% stability from body fixed
C_s_bf = [cos(alpha), 0, sin(alpha);
          0, 1, 0;
          -sin(alpha), 0, cos(alpha)];
% wind from stability
C_w_s = [cos(beta), sin(beta), 0;
         -sin(beta), cos(beta), 0;
         0, 0, 1];
% wind from body fixed
C_w_bf = C_w_s*C_s_bf;
% starting V
vw = [v;0;0];
vbf = C_w_bf'*vw;

% i
```

---

```

fprintf("i. Gust of 20 ft/s left to right (+velocity in y body fixed)\n")
vbf1 = vbf + [0;20;0];
disp("Don't know why this needs to be here")
[alpha1, beta1] = find_alpha_beta(vbf1);
disp("but it won't publish without these strings")
fprintf("\tAngle of attack is now %f degrees\n", alpha1)
fprintf("\tAngle of sideslip is now %f degrees\n", beta1)
% ii
fprintf("ii. Gust of 50 ft/s from dead astern(+velocity in x body
fixed)\n")
vbf2 = vbf + [50;0;0];
[alpha2, beta2] = find_alpha_beta(vbf2);
fprintf("\tAngle of attack is now %f degrees\n", alpha2)
fprintf("\tAngle of sideslip is now %f degrees\n", beta2)

% iii
fprintf("iii. Gust of 30 ft/s from right and below (-velocity in y, -
velocity in z\n")
vbf3 = vbf + [0;-30*cos(70/r2d);-30*sin(70/r2d)];
[alpha3, beta3] = find_alpha_beta(vbf3);
fprintf("\tAngle of attack is now %f degrees\n", alpha3)
fprintf("\tAngle of sideslip is now %f degrees\n", beta3)

```

\*\*\*\*\*Problem 3\*\*\*\*\*

*i. Gust of 20 ft/s left to right (+velocity in y body fixed)*

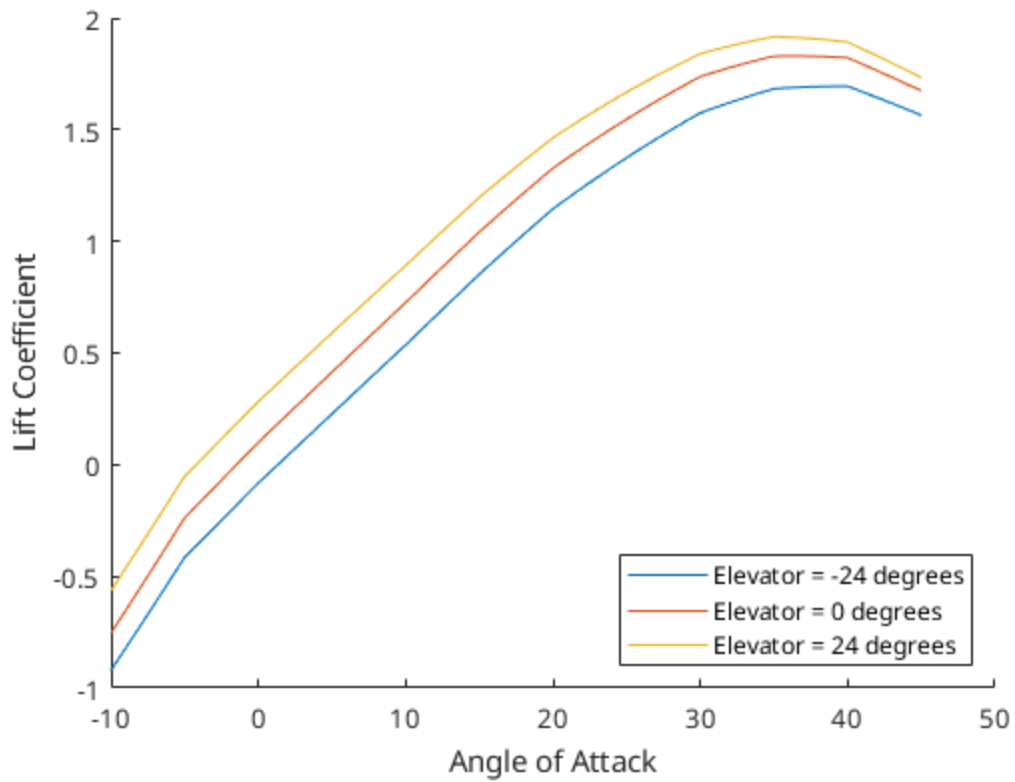
*Don't know why this needs to be here*

## Problem 4

problem\_4

\*\*\*\*\*Problem 4\*\*\*\*\*

*Max lift occurs at an angle of attack of 35.000000 degrees*

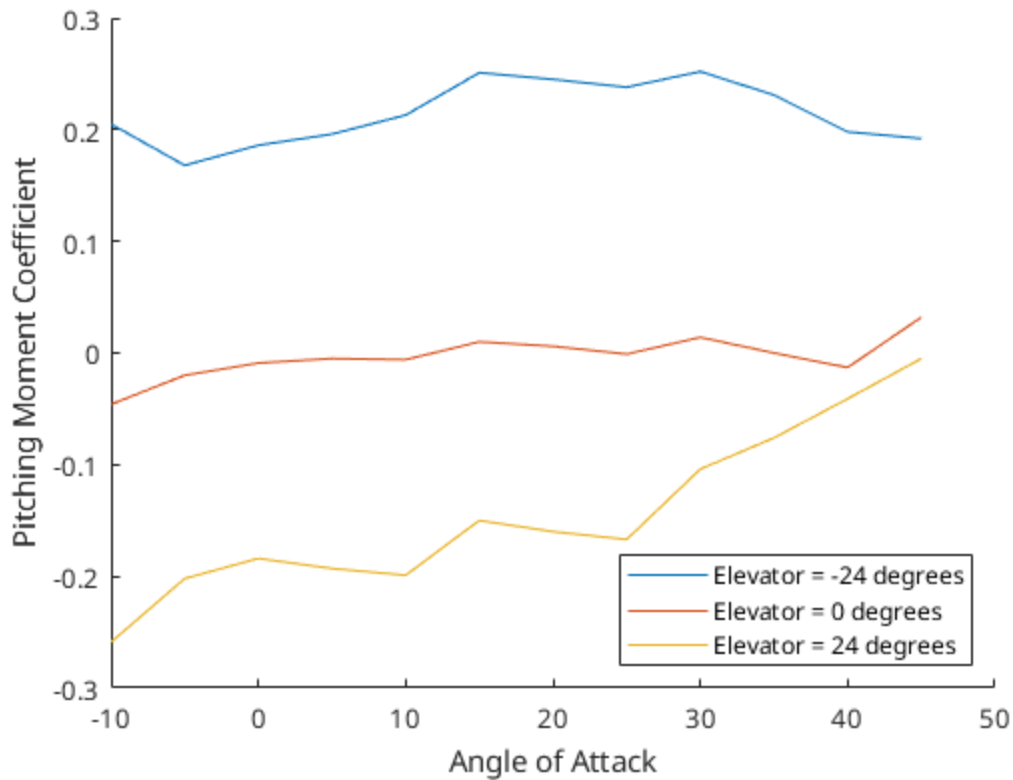


## Problem 5

problem\_5

\*\*\*\*\*Problem 5\*\*\*\*\*

*We can see that the F16 lacks pitch stiffness as the curves all vary from flat to a slight slope up. The elevator does significantly shift the whole curve showing it has significant control power*



## Problem 6

problem\_6

\*\*\*\*\*Problem 6\*\*\*\*\*

*I did not see a what pitch or flight path was used for Table 3.6-1  
so I adjusted pitch to match initial cost in the table*

*TAS: 170.000000 ft/s AOA: 0.385718 radians Pitch: 0.217817 radians  
Pitch Rate: 0.000000 radians/s Altitude: 0.000000 ft Distance 0.000000 ft  
Sum of Weighted Squares: 28.937859*

*TAS: 500.000000 ft/s AOA: 0.010123 radians Pitch: 0.068749 radians  
Pitch Rate: 0.000000 radians/s Altitude: 0.000000 ft Distance 0.000000 ft  
Sum of Weighted Squares: 3.542947*

*TAS: 500.000000 ft/s AOA: 0.094771 radians Pitch: 0.105418 radians  
Pitch Rate: 0.000000 radians/s Altitude: 0.000000 ft Distance 0.000000 ft  
Sum of Weighted Squares: 10.812971*

\*\*\*\*Find derivations for state equations\*\*\*\*

## Problem 7

problem\_7

\*\*\*\*\*Problem 7\*\*\*\*\*

---

Equilibrium Points:

```
x1=-2    x2=0
x1=1 - 3^(1/2)*1i    x2=0
x1=3^(1/2)*1i + 1    x2=0
x1=0    x2=-2*2^(1/2)
x1=0    x2=2*2^(1/2)
```

Linearize around  $x_1=-2$ ,  $x_2=0$  using first order Taylor series

The A matrix is:

```
[12,  0]
[ 0, -2]
```

The eigenvalues of this A matrix are  $\lambda = -2.000000$  and  $12.000000$

Linearize around  $x_1=1-3^{1/2}i$ ,  $x_2=0$  using first order Taylor series

The A matrix is:

```
[3*(- 1 + 3^(1/2)*1i)^2,    0]
[    0, 1 - 3^(1/2)*1i]
```

The eigenvalues of this A matrix are  $\lambda = 1.000000+-1.732051i$  and  $-6.000000+-10.392305i$

Linearize around  $x_1=1+3^{1/2}i$ ,  $x_2=0$  using first order Taylor series

The A matrix is:

```
[3*(1 + 3^(1/2)*1i)^2,    0]
[    0, 1 + 3^(1/2)*1i]
```

The eigenvalues of this A matrix are  $\lambda = 1.000000+1.732051i$  and  $-6.000000+10.392305i$

Linearize around  $x_1=0$ ,  $x_2=2*2^{1/2}$  using first order Taylor series

The A matrix is:

```
[    0, -4*2^(1/2)]
[2*2^(1/2),    0]
```

The eigenvalues of this A matrix are  $\lambda = 0.000000+-4.000000i$  and  $0.000000+4.000000i$

Linearize around  $x_1=0$ ,  $x_2=-2*2^{1/2}$  using first order Taylor series

The A matrix is:

```
[    0, 4*2^(1/2)]
[-2*2^(1/2),    0]
```

The eigenvalues of this A matrix are  $\lambda = 0.000000+-4.000000i$  and  $0.000000+4.000000i$

## Problems

```
function problem_2()
    fprintf('*****Problem 2*****\n')
%     % xdot = A*x + B*u
%     A = [8, -2, 3;
%          0, 1, 4;
```

---

```

%       7, 9, 10];
%       B = [2; 1; 5];
%       % y = C*x+D*u
%       C = [2, 1, 3];
%       D = 0;
%
%       % Transfer function G(s)
fprintf("Transfer Function\n")
%       syms s
%       G = collect(C*inv((s*eye(3)-A))*B);
%       fprintf("From formula\n")
%       fprintf("G(s) = %s\n", G)
%       fprintf("Built in Function\n")
%       [b,a] = ss2tf(A,B,C,D);
%       G_builtin = collect((b(1)*s^3 + b(2)*s^2 + b(3)*s + b(4))/...
%       (a(1)*s^3 + a(2)*s^2 + a(3)*s + a(4)));
%       fprintf("G(s) = %s\n", G_builtin)
%       % Poles
%       fprintf("\nPoles\n")
%       poles_builtin = eig(A);
%       poles = vpasolve(det(s*eye(3)-A)==0,s);
%       fprintf("From formula\n")
%       fprintf("Poles %f, %f, %f \n", poles)
%       fprintf("Built in Function\n")
%       fprintf("Poles %f, %f, %f \n", poles_builtin)
end

function problem_3()
    global r2d
    fprintf("\n*****Problem 3*****\n")
    % 2.3-1 Find instantaneous angle of attack and angle of side slip
    % in the gusts

    % Given starting conditions
    v = 500;
    alpha = 8/r2d;
    beta = -5/r2d;
    % Rotation matrices
    % stability from body fixed
    C_s_bf = [cos(alpha), 0, sin(alpha);
              0, 1, 0;
              -sin(alpha), 0, cos(alpha)];
    % wind from stability
    C_w_s = [cos(beta), sin(beta), 0;
             -sin(beta), cos(beta), 0;
             0, 0, 1];
    % wind from body fixed
    C_w_bf = C_w_s*C_s_bf;
    % starting V
    vw = [v;0;0];
    vbf = C_w_bf'*vw;

    % i
    fprintf("i. Gust of 20 ft/s left to right (+velocity in y body fixed)\n")

```

---

---

```

    vbf1 = vbf + [0;20;0];
    disp("Don't know why this needs to be here")
    [alpha1, beta1] = find_alpha_beta(vbf1);
    disp("but it won't publish without these strings")
    fprintf("\tAngle of attack is now %f degrees\n", alpha1)
    fprintf("\tAngle of sideslip is now %f degrees\n", beta1)
    % ii
    fprintf("ii. Gust of 50 ft/s from dead astern(+velocity in x body
fixed)\n")
    vbf2 = vbf + [50;0;0];
    [alpha2, beta2] = find_alpha_beta(vbf2);
    fprintf("\tAngle of attack is now %f degrees\n", alpha2)
    fprintf("\tAngle of sideslip is now %f degrees\n", beta2)

    % iii
    fprintf("iii. Gust of 30 ft/s from right and below (-velocity in y, -
velocity in z\n")
    vbf3 = vbf + [0;-30*cos(70/r2d);-30*sin(70/r2d)];
    [alpha3, beta3] = find_alpha_beta(vbf3);
    fprintf("\tAngle of attack is now %f degrees\n", alpha3)
    fprintf("\tAngle of sideslip is now %f degrees\n", beta3)

end

function problem_4()
    fprintf("\n*****Problem 4*****\n")
    % 2.3-4 Find instantaneous angle of attack and angle of side slip
    % in the gusts
    n = 1e2;
    alpha = linspace(-10,45,n);
    elev = [-24,0,24];
    for ii = 1:3
        cl(ii,:) = zeros(1,n);
        for jj = 1:n
            cl(ii,jj) = CX(alpha(jj),elev(ii))*sind(alpha(jj)) - ...
                CZ(alpha(jj),0,elev(ii))*cosd(alpha(jj));
        end
    end
    figure
    hold on
    plot(alpha, cl(1,:))
    plot(alpha, cl(2,:))
    plot(alpha, cl(3,:))
    legend("Elevator = -24 degrees", "Elevator = 0 degrees", ...
        "Elevator = 24 degrees", "Location","southeast")
    xlabel("Angle of Attack")
    ylabel("Lift Coefficient")
    hold off

    [~, I] = max(cl(3,:));
    fprintf("Max lift occurs at an angle of attack of %f degrees\n", alpha(I))
end

function problem_5()

```

---

---

```

fprintf("\n*****Problem 5*****\n")
% 2.3-5
n = 1e2;
alpha = linspace(-10,45,n);
elev = [-24,0,24];
for ii = 1:3
    cm(ii,:) = zeros(1,n);
    for jj = 1:n
        cm(ii,jj) = CM(alpha(jj),elev(ii));
    end
end
figure
hold on
plot(alpha, cm(1,:))
plot(alpha, cm(2,:))
plot(alpha, cm(3,:))
legend("Elevator = -24 degrees", "Elevator = 0 degrees", ...
    "Elevator = 24 degrees", "Location","southeast")
xlabel("Angle of Attack")
ylabel("Pitching Moment Coefficient")
hold off
fprintf("We can see that the F16 lacks pitch stiffness as the curves all
\n")
fprintf("vary from flat to a slight slope up. The elevator does
significantly \n")
fprintf("shift the whole curve showing it has significant control power
\n")
end

function problem_6()
    fprintf("\n*****Problem 6*****\n")
    global r2d
    % 3.5-1
    fprintf("I did not see a what pitch or flight path was used for Table
3.6-1\n")
    fprintf("so I adjusted pitch to match initial cost in the table\n")
    x1 = [170; 22.1/r2d; 12.48/r2d; 0; 0; 0];
    u1 = [.297, -25.7, .25, 0];
    [xd1] = transp(x1,u1);
    sos1 = xd1(1)^2 + 100*xd1(2)^2 + 10*xd1(4)^4;
    fprintf("TAS: %f ft/s AOA: %f radians Pitch: %f radians \n", x1(1:3))
    fprintf("Pitch Rate: %f radians/s Altitude: %f ft Distance %f ft\n",
x1(4:6))
    fprintf("\tSum of Weighted Squares: %f\n", sos1)

    x2 = [500; .58/r2d; 3.939/r2d; 0; 0; 0];
    u2 = [.293, 2.46, .25, 0];
    [xd2] = transp(x2,u2);
    sos2 = xd2(1)^2 + 100*xd2(2)^2 + 10*xd2(4)^4;
    fprintf("TAS: %f ft/s AOA: %f radians Pitch: %f radians \n", x2(1:3))
    fprintf("Pitch Rate: %f radians/s Altitude: %f ft Distance %f ft\n",
x2(4:6))
    fprintf("\tSum of Weighted Squares: %f\n", sos2)

```

---



---

```

x3 = [500; 5.43/r2d; 6.04/r2d; 0; 0; 0];
u3 = [.204, -4.10, .25, 0];
[xd3] = transp(x3,u3);
sos3 = xd3(1)^2 + 100*xd3(2)^2 + 10*xd3(4)^4;
fprintf("TAS: %f ft/s  AOA: %f radians  Pitch: %f radians  \n", x3(1:3))
fprintf("Pitch Rate: %f radians/s  Altitude: %f ft  Distance %f ft\n",
x3(4:6))
fprintf("\tSum of Weighted Squares: %f\n", sos3)

fprintf("****Find derivations for state equations****\n")

end

function problem_7()
    fprintf("\n*****Problem 7*****\n")
    global r2d
    % 3.7-1
    syms x1 x2
    xldot = x1^3-x2^2+8;
    x2dot = x1*x2;
    sol = solve(xldot==0,x2dot==0,x1,x2);
    fprintf("Equilibrium Points:\n")
    for ii = 1:5
        fprintf("\tx1=%s    x2=%s\n",[sol.x1(ii),sol.x2(ii)])
    end
    fprintf("Linearize around x1=-2, x2=0 using first order Taylor series\n")
    A = subs([diff(xldot,x1), diff(xldot,x2); diff(x2dot,x1), diff(x2dot,x2)],
[x1,x2],[-2,0]);
    fprintf("The A matrix is:\n")
    disp(A)
    [~,D] = eig(A);
    fprintf("\tThe eigenvalues of this A matrix are lambda= %f and %f\n\n",
[D(1,1),D(2,2)])

    fprintf("Linearize around x1=1-3^(1/2)*i, x2=0 using first order Taylor
series\n")
    A = subs([diff(xldot,x1), diff(xldot,x2); diff(x2dot,x1), diff(x2dot,x2)],
[x1,x2],[1-3^(1/2)*i,0]);
    fprintf("The A matrix is:\n")
    disp(A)
    [~,D] = eig(A);
    fprintf("\tThe eigenvalues of this A matrix are lambda= %f+%fi and %f+%fi
\n\n",[real(D(1,1)),imag(D(1,1)),real(D(2,2)),imag(D(2,2))])

    fprintf("Linearize around x1=1+3^(1/2)*i, x2=0 using first order Taylor
series\n")
    A = subs([diff(xldot,x1), diff(xldot,x2); diff(x2dot,x1), diff(x2dot,x2)],
[x1,x2],[1+3^(1/2)*i,0]);
    fprintf("The A matrix is:\n")
    disp(A)
    [~,D] = eig(A);
    fprintf("\tThe eigenvalues of this A matrix are lambda= %f+%fi and %f+%fi
\n\n",[real(D(1,1)),imag(D(1,1)),real(D(2,2)),imag(D(2,2))])

```

---

---

```

    fprintf("Linearize around x1=0, x2=2*2^(1/2) using first order Taylor
series\n")
    A = subs([diff(xldot,x1), diff(xldot,x2); diff(x2dot,x1), diff(x2dot,x2)],
[x1,x2],[0,2*2^(1/2)]);
    fprintf("The A matrix is:\n")
    disp(A)
    [~,D] = eig(A);
    fprintf("\tThe eigenvalues of this A matrix are lambda= %f+%fi and %f+%fi
\n\n",[real(D(1,1)),imag(D(1,1)),real(D(2,2)),imag(D(2,2))])

    fprintf("Linearize around x1=0, x2=-2*2^(1/2) using first order Taylor
series\n")
    A = subs([diff(xldot,x1), diff(xldot,x2); diff(x2dot,x1), diff(x2dot,x2)],
[x1,x2],[0,-2*2^(1/2)]);
    fprintf("The A matrix is:\n")
    disp(A)
    [~,D] = eig(A);
    fprintf("\tThe eigenvalues of this A matrix are lambda= %f+%fi and %f+%fi
\n\n",[real(D(1,1)),imag(D(1,1)),real(D(2,2)),imag(D(2,2))])

end

```

## Functions

```

function [alpha, beta] = find_alpha_beta(vbf)
    global r2d
    syms alphas betas V
    C_s_bf = [cos(alphas), 0, sin(alphas);
              0, 1, 0;
              -sin(alphas), 0, cos(alphas)];
    % wind from stability
    C_w_s = [cos(betas), sin(betas), 0;
              -sin(betas), cos(betas), 0;
              0, 0, 1];
    % wind from body fixed
    C_w_bf = C_w_s*C_s_bf;
    sol = vpasolve([V;0;0]==C_w_bf*vbf,[alphas,betas,V]);
    alpha = sol.alphas*r2d;
    beta = sol.betas*r2d;

end

function [CXI]=CX(alpha,elev)
    % Data and interpolation for F-16 Axial force coefficient.
    alphatabl=[-10,-5,0,5,10,15,20,25,30,35,40,45];
    elevtabl=[-24,-12,0,12,24];
    cxatabl=[-.099, -.081, -.081, -.063, -.025, .044, .097...
             .113, .145, .167, .174, .166;
             -.048, -.038, -.040, -.021, .016, .083, .127...
             .137, .162, .177, .179, .167;
             -.022, -.020, -.021, -.004, .032, .094, .128...
             .130, .154, .161, .155, .138;
             -.040, -.038, -.039, -.025, .006, .062, .087...
             .085, .100, .110, .104, .091;

```

---

```

    -.083, -.073, -.076, -.072, -.046, .012, .024...
    .025, .043, .053, .047, .040];
    CXI= interp2(alphatabl,elevtabl,cxtabl,alpha,elev);
end

function [CZI]=CZ(alpha,beta,elev)
    % Data and interpolation for F-16 z-force coefficient
    alphatabl=[-10,-5,0,5,10,15,20,25,30,35,40,45];
    cztabl=[.770, .241, -.100, -.416, -.731, -1.053, -1.366...
    -1.646, -1.917, -2.120, -2.248, -2.229];
    C1= interp1(alphatabl,cztabl,alpha);
    CZI= C1*(1-(beta/57.3)^2) - .19*(elev/25.0);
end

function [CMI]=CM(alpha,elev)
    % F-16 model pitching-moment data and interpolation
    alphatabl=[-10,-5,0,5,10,15,20,25,30,35,40,45];
    elevtabl=[-24,-12,0,12,24];
    cmtabl=[.205, .168, .186, .196, .213, .251, .245...
    .238, .252, .231, .198, .192;
    .081, .077, .107, .110, .110, .141, .127...
    .119, .133, .108, .081, .093;
    -.046, -.020, -.009, -.005, -.006, .010, .006...
    -.001, .014, .000, -.013, .032;
    -.174, -.145, -.121, -.127, -.129, -.102, -.097...
    -.113, -.087, -.084, -.069, -.006;
    -.259, -.202, -.184, -.193, -.199, -.150, -.160...
    -.167, -.104, -.076, -.041, -.005];
    CMI = interp2(alphatabl,elevtabl,cmtabl,alpha,elev);
end

function [xd] = transp(x,u)
    % Medium-sized transport aircraft, longitudinal dynamics
    S = 2170.0; CBAR = 17.5; MASS = 5e3; IYY = 4.1e6;
    TSTAT = 6e4; DTDV = -38; ZE = 2; CDCLS = .042;
    CLA = .085; CMA = -.022; CMDE = -.016; % per degree
    CMQ = -16; CMADOT = -6; CLADOT = 0; % per radian
    RTOD = 57.29578; GD = 32.17;
    THTL = u(1);
    ELEV = u(2);
    XCG = u(3);
    LAND = u(4);
    VT = x(1); % TAS in fps
    ALPHA = RTOD*x(2); % A. O. A.
    THETA = x(3); % PITCH ATTITUDE
    Q = x(4); % PITCH RATE
    H = x(5); % ALTITUDE
    [~,QBAR] = ADC(VT,H);
    QS = QBAR*S;
    SALP = sin(x(2)); CALP = cos(x(2));
    GAM = THETA - x(2); SGAM = sin(GAM); CGAM = cos(GAM);
    if LAND == 0 % clean
        CLO = .2; CDO = .016;
        CMO = .05; DCDG = 0; DCMG = 0;
    end

```

---

---

```

elseif LAND == 1    % landing flap and gear
    CLO = 1; CDO = .08;
    CMO = -.2; DCDG = .02; DCMG = -.05;
else
    disp("Landing Gear & Flaps")
end
THR = (TSTAT+VT*DTDV) * max(THTL,0);           % THR
CL = CLO+CLA*ALPHA;                             % NONDIM LIFT
CM = DCMG+CMO+CMA*ALPHA+CMDE*ELEV+CL*(XCG-.25); % MOMENT
CD = DCDG+CDO+CDCLS*CL*CL;                     % DRAG POLAR

% State Equations
xd(1) = (THR*CALP-QS*CD)/MASS - GD*SGAM;
xd(2) = (-THR*SALP-QS*CL+MASS*(VT*Q+GD*CGAM))/(MASS*VT+QS*CLADOT);
xd(3) = Q;
D = .5*CBAR*(CMQ*Q+CMADOT*xd(2))/VT;
xd(4) = (QS*CBAR*(CM+D)+THR*ZE)/IYY;
xd(5) = VT*SGAM;
xd(6) = VT*CGAM;
end

function [MACH,QBAR] = ADC(VT,H)
    R0 = 2.377e-3;
    TFAC = 1-.703e-5*H;
    T = 519*TFAC;
    if H >= 35000
        T = 390;
    end
    RHO = R0*(TFAC^4.14);
    MACH = VT/sqrt(1.4*1716.3*T);
    QBAR = .5*RHO*VT*VT;
end

but it won't publish without these strings
Angle of attack is now 8.000000 degrees
Angle of sideslip is now -2.710123 degrees
ii. Gust of 50 ft/s from dead astern(+velocity in x body fixed)
Angle of attack is now 7.271964 degrees
Angle of sideslip is now -4.549540 degrees
iii. Gust of 30 ft/s from right and below (-velocity in y, -velocity in z)
Angle of attack is now 4.766735 degrees
Angle of sideslip is now -6.207827 degrees

```

*Published with MATLAB® R2021b*