Problem 1

```
disp('');


function [trim_state, trim_control] = TrimVariableToState(trim_variable,
trim_definition)
% Calculates the aircraft trim state vector and control surface vector
% Inputs:
%   trim_definition -> [Va; gamma0; h0] -> [air speed; air relative flight
path angle; height]
%   trim_variable -> [alpha0; dele0, delt0] [angle of attack; elevator;
%   throttle]
% Output:
%   trim_state -> full 12x1 aircraft state in trim
%   trim_control -> [de; da; dr; dt] -> [elevator; aileron; rudder;
%   throttle]
%
% Author: Thomas Dunnington
% Date Modified: 9/16/2024

% Get input variables
Va = trim_definition(1);
gamma0 = trim_definition(2);
h0 = trim_definition(3);

alpha = trim_variable(1);
elevator = trim_variable(2);
throttle = trim_variable(3);

% Pitch
theta = gamma0 + alpha;

% z position
z = -h0;

% u velocity
u = Va*cos(alpha);

% w velocity
w = Va*sin(alpha);

% Aicraft state vector
trim_state = [0;0;z;0;theta;0;u;0;w;0;0;0];

% Control Surface
trim_control = [elevator; 0; 0; throttle];

end


function [cost] = TrimCostFunction(trim_variable, trim_definition,
```

```
aircraft_parameters)
% Returns the cost function for the trim condition
% Inputs:
%   trim_definition -> [Va; gamma0; h0] -> [air speed; air relative flight
path angle; height]
%   trim_variable -> [alpha0; dele0, delt0] [angle of attack; elevator;
%   throttle]
%   aircraft_parameters -> Aircraft parameters structure
% Output:
%   cost -> Trim condition cost function
%
% Author: Thomas Dunnington
% Date Modified: 9/16/2024

% Get input variables
Va = trim_definition(1);
gamma0 = trim_definition(2);
h0 = trim_definition(3);

alpha = trim_variable(1);
elevator = trim_variable(2);
throttle = trim_variable(3);

% Assume zero wind
wind_inertial = [0;0;0];

% Air Density
density = stdatmo(h0);

% Calculate trim
[trim_state, trim_control] =
TrimVariableToState(trim_variable,trim_definition);

% Force Calculations
[aircraft_forces, aircraft_moments] = AircraftForcesAndMoments(trim_state,
trim_control, wind_inertial, density, aircraft_parameters);

% Cost
cost = norm(aircraft_forces) + norm(aircraft_moments);

end


function [trim_state,trim_control] = TrimCalculator(trim_definition,
aircraft_parameters)
% Uses fmincon to calculate the aircraft state and control surface
% variables
% Inputs:
%   trim_definition -> [Va; gamma0; h0] -> [air speed; air relative flight
path angle; height]
%   aircraft_parameters -> Aircraft parameters structure
%   wind_inertial -> inertial wind velocity vector in the inertial frame
% Output:
%   cost -> Trim condition cost function
```

```
%
% Author: Thomas Dunnington
% Date Modified: 9/16/2024

% Cost Function handle
cost_func = @(trim_variable)TrimCostFunction(trim_variable, trim_definition,
aircraft_parameters);

% Fmincon call
x0 = zeros(3,1);
[trim_variables_final, ~] = fmincon(cost_func, x0);

% Get trim state and control variables
[trim_state, trim_control] =
TrimVariableToState(trim_variables_final,trim_definition);
end
```

Problem 2

```
function [trim_state, trim_control] =
CoordinatedTurnVariableToState(trim_variable, trim_definition)
% Calculates the aircraft trim state vector and control surface vector
% Inputs:
%   trim_definition -> [Va; gamma0; h0] -> [air speed; air relative flight
path angle; height]
%   trim_variable -> [alpha0; dele0, delt0] [angle of attack; elevator;
%   throttle]
% Output:
%   trim_state -> full 12x1 aircraft state in trim
%   trim_control -> [de; da; dr; dt] -> [elevator; aileron; rudder;
%   throttle]
%
% Author: Thomas Dunnington
% Date Modified: 9/16/2024

% Get input variables
Va = trim_definition(1);
gamma0 = trim_definition(2);
h0 = trim_definition(3);
R0 = trim_definition(4);

alpha = trim_variable(1);
elevator = trim_variable(2);
throttle = trim_variable(3);
roll = trim_variable(4);
beta = trim_variable(5);
aileron = trim_variable(6);
rudder = trim_variable(7);

% Position
pos_vec = [0; 0; -h0];

% Euler Angles
```

```
phi = roll;
theta = gamma0 + alpha;
psi = 0;
euler_anlges = [phi; theta; psi];

% Velocity vector
vel_vec = Va .* [cos(alpha)*cos(beta); sin(beta); sin(alpha)*cos(beta)];

% Angular velocity
chi_dot = Va / R0;
omega = TransformFromInertialToBody([0;0;chi_dot], euler_anlges);
%omega = [-sin(theta); sin(phi)*cos(theta); cos(phi)*cos(theta)] .* chi_dot;

% Aicraft state vector
trim_state = [pos_vec; euler_anlges; vel_vec; omega];

% Control Surface
trim_control = [elevator; aileron; rudder; throttle];
end


function [cost] = CoordinatedTurnCostFunction(trim_variable,
trim_definition, aircraft_parameters)
% Returns the cost function for a coordinated turn
% Inputs:
%   trim_definition -> [Va; gamma0; h0] -> [air speed; air relative flight
path angle; height]
%   trim_variable -> [alpha0; dele0, delt0] [angle of attack; elevator;
%   throttle]
%   aircraft_parameters -> Aircraft parameters structure
% Output:
%   cost -> Coordinate turn cost function
%
% Author: Thomas Dunnington
% Date Modified: 9/16/2024

% Get input variables
Va = trim_definition(1);
gamma0 = trim_definition(2);
h0 = trim_definition(3);
R0 = trim_definition(4);

alpha = trim_variable(1);
elevator = trim_variable(2);
throttle = trim_variable(3);
roll = trim_variable(4);
beta = trim_variable(5);
aileron = trim_variable(6);
rudder = trim_variable(7);

% Air Density
density = stdatmo(h0);

% Assume zero wind
```

```
wind_inertial = [0;0;0];

% Calculate trim
[trim_state, trim_control] =
CoordinatedTurnVariableToState(trim_variable,trim_definition);

% Force Calculations
[aircraft_forces, aircraft_moments] = AircraftForcesAndMoments(trim_state,
trim_control, wind_inertial, density, aircraft_parameters);

% Acceleration in the inertial Y direction
fdes_inertial = aircraft_parameters.m * [0; Va*Va/R0; 0];

% Rotate to body
fdes_body = TransformFromInertialToBody(fdes_inertial, trim_state(4:6));

% Total force
sum_forces = aircraft_forces - fdes_body;

% Calculate aerodynamic moments
[aero_force, aero_moment] =
AeroForcesAndMoments_BodyState_WindCoeffs(trim_state, trim_control,
wind_inertial, density, aircraft_parameters);

% Cost
cost = norm(sum_forces) + norm(aircraft_moments) + aero_force(2)^2;

end


function [trim_state,trim_control] =
CoordinatedTurnCalculator(trim_definition, aircraft_parameters)
% Uses fmincon to calculate the aircraft state and control surface
% variables
% Inputs:
%   trim_definition -> [Va; gamma0; h0] -> [air speed; air relative flight
path angle; height]
%   aircraft_parameters -> Aircraft parameters structure
%   wind_inertial -> inertial wind velocity vector in the inertial frame
% Output:
%   cost -> Trim condition cost function
%
% Author: Thomas Dunnington
% Date Modified: 9/16/2024

% Cost Function handle
cost_func = @(trim_variable)CoordinatedTurnCostFunction(trim_variable,
trim_definition, aircraft_parameters);

% Fmincon call
x0 = zeros(7,1);
[trim_variables_final, fval] = fmincon(cost_func, x0);

% Get trim state and control variables
```

```
[trim_state, trim_control] =
CoordinatedTurnVariableToState(trim_variables_final, trim_definition);
end
```

Published with MATLAB® R2023b