```
clear all
clc

%-------------------------------------------------------------------------%
%------------------ Provide Auxiliary Data for Problem -------------------%
%-------------------------------------------------------------------------%
auxdata.rho  = 1.225;
auxdata.CD0  = 0.00873;
auxdata.K    = 0.045;
auxdata.g    = 9.81;
auxdata.m    = 10;
auxdata.S    = 1;
% % % auxdata.beta = 0.063726299384995;
auxdata.beta = 0.1;
% % % auxdata.beta = 0.08;
auxdata.W0   = 0;
auxdata.lmin = -2;
auxdata.lmax = 5;
auxdata.A    = 1.1;

%-------------------------------------------------------------------------%
%---------------------- Boundary Conditions ------------------------------%
%-------------------------------------------------------------------------%
t0 = 0; x0 = 0; y0 = 0; z0 = 10; v0 = 100; A0 = 0;

%-------------------------------------------------------------------------%
%---------------------- Limits on Variables ------------------------------%
%-------------------------------------------------------------------------%
tfmin    = 1;          tfmax    = 1000;
xmin     = -1000;      xmax     = +1000;
ymin     = -1000;      ymax     = +1000;
zmin     =  10;        zmax     = +1000;
vmin     = +10;        vmax     = +350;
gammamin = -75*pi/180; gammamax = 75*pi/180;
psimin   = -3*pi;      psimax   = +pi/2;
% % % Amin     = 0;          Amax     = 1e10;
CLmin    = -2;         CLmax    = 1.5;
mumin    = -180/180*pi;mumax    = 75/180*pi;
% % % beta_min = 0.005;      beta_max = 0.15;

k_max = (0.5*auxdata.rho.*auxdata.S.*CLmax) / auxdata.m;
k_min = (0.5*auxdata.rho.*auxdata.S.*CLmin) / auxdata.m;

%-------------------------------------------------------------------------%
%--------------- Set Up Problem Using Data Provided Above ----------%
%-------------------------------------------------------------------------%
bounds.phase.initialtime.lower  = t0;
bounds.phase.initialtime.upper  = t0;
bounds.phase.finaltime.lower    = tfmin;
bounds.phase.finaltime.upper    = tfmax;
bounds.phase.initialstate.lower = [x0, y0, z0, vmin, gammamin, psimin];
```

```
bounds.phase.initialstate.upper = [x0, y0, z0, vmax, gammamax, psimax];
bounds.phase.state.lower        = [xmin, ymin, zmin, vmin, gammamin, psimin];
bounds.phase.state.upper        = [xmax, ymax, zmax, vmax, gammamax, psimax];
bounds.phase.finalstate.lower   = [x0, y0, z0, vmin, gammamin, psimin];
bounds.phase.finalstate.upper   = [x0, y0, z0, vmax, gammamax, psimax];
bounds.phase.control.lower      = [CLmin, mumin];
bounds.phase.control.upper      = [CLmax, mumax];
bounds.phase.path.lower         = [auxdata.lmin , k_min];
bounds.phase.path.upper         = [auxdata.lmax , k_max];
% % % bounds.phase.integral.lower     = -1e15;
bounds.phase.integral.lower     = 0;
bounds.phase.integral.upper     = 1e15;
bounds.eventgroup.lower         = [0, 0, -2*pi];
bounds.eventgroup.upper         = [0, 0, -2*pi];
% % % bounds.parameter.lower          = beta_min;
% % % bounds.parameter.upper          = beta_max;

%-------------------------------------------------------------------------%
%---------------------- Provide Guess of Solution ------------------------%
%-------------------------------------------------------------------------%
N                               = 100;
CL0                             = CLmax;
tGuess                          = linspace(0,24,N).';
xguess                          = 500*cos(2*pi*tGuess/24)-500;
yguess                          = 300*sin(2*pi*tGuess/24);
zguess                          = -400*cos(2*pi*tGuess/24)+400;
vguess                          = 0.8*v0*(1.5+cos(2*pi*tGuess/24));
gammaguess                      = pi/6*sin(2*pi*tGuess/24);
psiguess                        = -1-tGuess/4;
CLguess                         = CL0*ones(N,1)/3;
muguess                         = -ones(N,1);
% % % beta_guess                     = 0.08;
guess.phase.time                = tGuess;
guess.phase.state               = [xguess, yguess, zguess, vguess, gammaguess, psiguess];
guess.phase.control             = [CLguess, muguess];
% % % guess.parameter                = beta_guess;
guess.phase.integral            = 1e+02;

%-------------------------------------------------------------------------%
%----------Provide Mesh Refinement Method and Initial Mesh ---------------%
%-------------------------------------------------------------------------%
mesh.maxiteration               = 10;
mesh.method                     = 'hp-LiuRao';
mesh.tolerance                  = 1e-5;
% % % mesh.tolerance                 = 1e-6;
% % % mesh.tolerance                 = 1e0;

%-------------------------------------------------------------------------%
%---------- Configure Setup Using the information provided ---------------%
%-------------------------------------------------------------------------%
setup.name                              = 'Dynamic-Soaring-Problem';
```

```
setup.functions.continuous              = @dynamicSoaringContinuous;
setup.functions.endpoint                = @dynamicSoaringEndpoint;
setup.nlp.solver                        = 'ipopt';
setup.nlp.ipoptoptions.linear_solver    = 'mumps';
setup.nlp.ipoptoptions.tolerance        = 1e-7;
setup.nlp.ipoptoptions.maxiterations    = 2e8;
setup.nlp.ipoptoptions.linear_solver    = 'ma57';
setup.displaylevel                      = 2;
setup.auxdata                           = auxdata;
setup.bounds                            = bounds;
setup.guess                             = guess;
setup.mesh                              = mesh;
setup.derivatives.supplier              = 'sparseCD';
setup.derivatives.derivativelevel       = 'second';
setup.scales.method                     = 'none';
setup.method                            = 'RPM-Differentiation';

%---------------------------------------------------------------------%
%------------------ Solve Problem Using GPOPS-II--------------------%
%---------------------------------------------------------------------%
output = gpops2(setup);
solution = output.result.solution;

% % % dynamicSoaringPlot;
multi_axes;
t_f = max(solution.phase.time)
avg_Minimum_Difference_radii = solution.phase.integral/ max(solution.phase.time)
```